

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-175210

(43)Date of publication of application : 21.06.2002

(51)Int.Cl.

G06F 12/00

G06F 12/14

G09C 1/00

(21)Application number : 2000-373593

(71)Applicant : TOYO COMMUN EQUIP CO LTD

(22)Date of filing : 07.12.2000

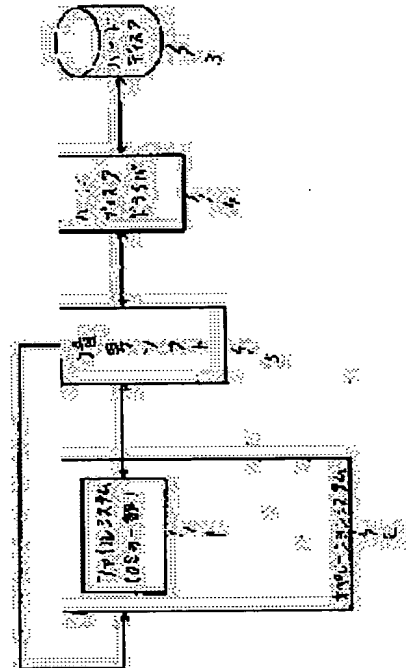
(72)Inventor : ICHISE HIROSHI

(54) METHOD OF TRANSFERRING AND COPYING, AND ENCIPHERING AND DECIPHERING DATA

(57)Abstract:

PROBLEM TO BE SOLVED: To enhance processing stability and a security function.

SOLUTION: This method is provided with a step for obtaining a control function for controlling a device driver such as a hard disk, a step for opening an encipherment file using the control function when detecting that the file within a cipher area is transferred or copied to a noncipher area, a step for reading the opened encipherment file using the control function to generate an ordinary sentence file by deciphering, a step for opening the file in a transferred site or copied site using the control function, a step for writing the ordinary sentence file in the transferred site or copied site using the control function, a step for closing the ordinary sentence file in the transferred site or copied site using the control function, and a step for deleting the encipherment file within the cipher area using the control function in the case of the transfer.



(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号
特開2002-175210
(P2002-175210A)

(43)公開日 平成14年6月21日(2002.6.21)

(51)Int.Cl. ⁷	識別記号	F I	テーマコード(参考)
G 0 6 F 12/00	5 3 7	G 0 6 F 12/00	5 3 7 H 5 B 0 1 7
	5 1 5		5 1 5 M 5 B 0 8 2
12/14	3 2 0	12/14	3 2 0 B 5 J 1 0 4
G 0 9 C 1/00	6 6 0	G 0 9 C 1/00	6 6 0 D

審査請求 未請求 請求項の数4 O L (全 17 頁)

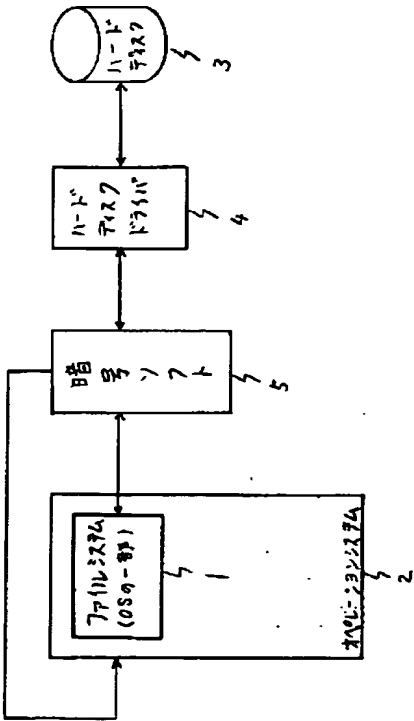
(21)出願番号	特願2000-373593(P2000-373593)	(71)出願人	000003104 東洋通信機株式会社 神奈川県高座郡寒川町小谷2丁目1番1号
(22)出願日	平成12年12月7日(2000.12.7)	(72)発明者	市瀬 浩 神奈川県高座郡寒川町小谷二丁目1番1号 東洋通信機株式会社内
		Fターム(参考)	5B017 AA03 BA07 CA16 5B082 AA11 GA02 GC05 5J104 AA01 AA08 AA13 AA16 EA04 EA26 LA05 NA02 NA12

(54)【発明の名称】 データの移動、複製方法及び暗号化、復号方法

(57)【要約】 (修正有)

【課題】処理安定度が高く、且つセキュリティ機能の高い暗号ソフトのデータの移動、複製方法及び暗号化、復号方法。

【解決手段】ハードディスク等のデバイスドライバをコントロールする為の制御関数を取得するステップと、暗号領域内の暗号化ファイルが非暗号領域への移動又は複製が実行されることを検知すると前記制御関数を用いて当該暗号化ファイルをオープンするステップと、オープンした暗号化ファイルを前記制御関数を用いて読み出し、これを復号して平文ファイルを生成するステップと、移動先又は複製先にて前記制御関数を用いてファイルをオープンするステップと、前記平文ファイルを前記制御関数を用いて移動先又は複製先に書き込むステップと、移動先又は複製先の前記平文ファイルを前記制御関数を用いてクローズするステップと、移動の場合、暗号領域内の前記暗号化ファイルを前記制御関数を用いて削除するステップとを備えた。



【特許請求の範囲】

【請求項1】 コンピュータシステムの特定のフォルダを暗号領域と設定し、該暗号領域内に格納された暗号化ファイルを、復号した上で前記フォルダ以外の非暗号領域に移動又は複製する方法であって、ハードディスク等のデバイスドライバをコントロールする為の制御関数を取得するステップと、暗号領域内の暗号化ファイルが非暗号領域への移動又は複製が実行されることを検知すると前記制御関数を用いて当該暗号化ファイルをオープンするステップと、オープンした暗号化ファイルを前記制御関数を用いて読み出し、これを復号して平文ファイルを生成するステップと、移動先又は複製先にて前記制御関数を用いてファイルをオープンするステップと、前記平文ファイルを前記制御関数を用いて移動先又は複製先に書き込むステップと、移動先又は複製先の前記平文ファイルを前記制御関数を用いてクローズするステップと、移動の場合、暗号領域内の前記暗号化ファイルを前記制御関数を用いて削除するステップとを備えたことを特徴とするデータの移動、複製方法及び暗号化、復号方法。

【請求項2】 コンピュータシステムの特定のフォルダを暗号領域と設定し、前記フォルダ以外の非暗号領域内に格納された平文ファイルを、暗号化した上で前記暗号領域に移動又は複製する方法であって、ハードディスク等のデバイスドライバをコントロールする為の制御関数を取得するステップと、非暗号領域内の平文ファイルが暗号領域への移動又は複製が実行されることを検知すると前記制御関数を用いて当該平文ファイルをオープンするステップと、オープンした平文ファイルを前記制御関数を用いて読み出し、これを暗号化して暗号化ファイルを生成するステップと、移動先又は複製先にて前記制御関数を用いてファイルをオープンするステップと、前記暗号化ファイルを前記制御関数を用いて移動先又は複製先に書き込むステップと、移動先又は複製先の前記暗号化ファイルを前記制御関数を用いてクローズするステップと、移動の場合、非暗号領域内の前記平文ファイルを前記制御関数を用いて削除するステップとを備えたことを特徴とするデータの移動、複製方法及び暗号化、復号方法。

【請求項3】 前記移動する対象がフォルダであった場合、当該フォルダ内のデータを順次請求項1又は請求項2に記載の方法にて移動する第1のステップと、前記データがサブフォルダであった場合、該サブフォルダより上位のフォルダ内の他のデータに先立って当該サブフォルダ内のデータを順次請求項1又は請求項2に記載の方法にて移動する第2のステップと、前記第2のステップを繰り返し、移動すべきデータが尽きると移動対象の最上位フォルダに戻って、前記第1のステップと第2のステップとを移動対象のフォルダ内のデータが全て移動するまで実行することを特徴とするデータの移動、複製方法及び暗号化、復号方法。

【請求項4】 コンピュータシステムにてデータを暗号化する、若しくは暗号化されたデータを復号する方法であって、事前に使用者を設定するために使用者に識別符号の入力を促すステップと、入力された識別符号のメッセージダイジェストを生成するステップと、生成したメッセージダイジェストを複数のデータ列に分割するステップと、分割した前記データ列の少なくとも一つを当該使用者の認証データとして記憶するステップとを備え、所要のデータに対して暗号化又は復号を施すために使用者に識別符号のメッセージダイジェストを生成するステップと、生成されたメッセージダイジェストを複数のデータ列に分割するステップと、分割した前記データ列から前記認証データに相当するものを抽出するステップと、これを予め記憶した前記認証データと比較するステップと、認証データが一致したときに前記データ列のうち前記認証データ以外のデータ列の少なくとも一つを暗号鍵として暗号化又は復号を実施するステップとを備えたことを特徴とするデータの移動、複製方法及び暗号化、復号方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、パーソナルコンピュータシステム等において電子ファイルを暗号化或いは復号しながら電子ファイルを移動させる方法に関する。

【0002】

【従来の技術】 近年、パーソナルコンピュータ等で作成した画像データやテキストデータ等の電子ファイルを簡単に暗号化できる暗号ソフトが販売されている。例えばソフトウェアを起動すると、まず最初に所定のパスワードを入力して使用者の認定を行ない、次に所定のフォルダを暗号化領域に指定するタイプの暗号化ソフトが知られている。このソフトウェアをコンピュータ上に常駐させておくと、所定のアプリケーションで作成したテキストデータ等の電子ファイルを前記暗号領域に保存しようとする、暗号ソフトはこれを察知し前記電子ファイルを暗号化して保存するようになっている。また、暗号領域内の電子ファイルをマウスでダブルクリックすると、前記電子ファイルは復号して所定のアプリケーション

(例えばワープロソフト等) に処理を渡すようになっているので、画面上では電子ファイルが通常のテキストファイルとして表示されるようになっている。

【0003】 一方、マイクロソフト社のWindows 95等をOS(オペレーションシステム)とするコンピュータにおいては、ファイル管理用のツールとしてエクスプローラを標準搭載しており、ファイルの移動や複写を容易に実行することができる。ところが、このエクスプローラを用いて暗号化した電子ファイルを暗号領域から非暗号領域へ移動させた場合、前記電子ファイルはパス名(保存先フォルダ名)が変更されるだけでファイルは自動的に復号されない。更に通常の電子ファイルを非

暗号領域から暗号領域へと移動させても、やはりファイルのパス名が変更されるだけで自動的に暗号化されない。そこで、ファイル移動の操作を察知して自動的に電子ファイルを暗号化或いは復号してくれる機能を持った暗号ソフトが提案されている。

【0004】図13はパーソナルコンピュータシステムに従来の暗号ソフトをインストールした状態を簡単に示したブロック図である。図13において、パーソナルコンピュータにはファイルシステム1を有するOS（オペレーションシステム）2がインストールされているとともに、ハードディスク3を制御する為のハードディスクドライバ4がインストールされている。暗号ソフト5はオペレーションシステム2とハードディスクドライバ4を介してハードディスク3内の所定の電子ファイルにアクセス出来るようになっている。ここで、暗号化ソフト5の一部はオペレーションシステム2とハードディスクドライバ4間に介在している。なお、パーソナルコンピュータシステムとしてはマウス、キーボード、CRT等を備えているが説明を簡単にするために図示を省略する。

【0005】図14は図13に示した従来の暗号ソフトの動作例を示したフローチャート図である。以下従来の暗号ソフトの動作例を図14を使いながら説明する。まず、暗号ソフトが起動した状態で、ユーザーはCRT等の表示画面を見ながらマウスをドラッグ&ドロップ操作し、暗号領域として指定した所定のフォルダ内に所定のファイル（暗号化していない通常の電子ファイル）を移動させる。前記ファイル移動操作をオペレーティングシステム2は検知し、所定フォルダ（以下移動元フォルダと言う）内の所定ファイルを暗号領域として指定したフォルダ（以下移動先フォルダと言う）内へ移動するように移動要求をファイルシステム1に送出し、ファイルシステム1はこれを暗号ソフト5に送出する。

【0006】ここで、暗号ソフト5は前記移動要求に含まれる情報を常時監視している。もし、前記移動要求が非暗号領域内から暗号領域内への移動若しくは暗号領域内から非暗号領域内への移動を指示するものであれば、暗号ソフト5はこの移動要求をハードディスクドライバ4に引き渡さず、移動が完了したことを示す移動完了応答をファイルシステム1を介してオペレーションシステム2に返送し、続いて図12で示した暗号処理をオペレーションシステム2を介し実行するようになっている。これは、オペレーションシステム2は前記移動要求に対し移動完了応答が返送されないと次の処理を受付けないようになっているからである。よって、暗号ソフト5は前記移動要求に対して、移動完了応答をオペレーションシステム5に返送し、別途図12の処理をオペレーションシステム5に指示するようになっている。

【0007】そこで、図12に示すように暗号ソフト5は、まず前記移動要求に含まれる情報から移動元ファイ

ルの属性（移動元ファイルのパス、日付等の情報）を読み出すようにオペレーションシステム2（以下OSと言う）に指示する。そして、入手した移動元ファイルの属性を基に、移動元ファイルをオープン（開く）ようにOSに指示すると共に、移動元ファイルをハードディスク3から読み込むようOSに指示する。次に、暗号ソフト5はハードディスク3から読み込んだ移動元ファイルを暗号化し、これをハードディスク3内に移動先ファイルとして書き込むようOSに指示する。書き込みが終了すると、暗号ソフト5は開いていた移動元ファイルをクローズ（閉じる）様にOSに指示し、更に開いていた移動先ファイルをクローズ（閉じる）様OSに指示する。

【0008】更に、暗号ソフト5はハードディスク3に保存した移動先のファイル属性でファイル日付を修正するようにOSに指示する。これは、移動元ファイルを暗号化し移動先ファイルに書き込んだ時点で、ファイルの日付が更新されてしまうのを防止する目的で実行している。そして、暗号ソフト5は移動元ファイルを削除する様にOSに指示を出し、最終的に移動元ファイルはハードディスク3から削除される。このように、図12に示した一連の処理によって、移動元ファイルは自動的に暗号化された上で移動先ファイルに複製され、更に移動元ファイルは削除されるので、あたかもファイルが暗号化されて移動しているように処理されたことになる。なお、暗号領域から非暗号領域へファイルを移動させた場合も図12に示すように、移動元ファイルは自動的に復号されて移動しているように処理される。このように、暗号ソフト5は暗号化／復号する処理以外は全てOSに指示を与えて処理を実行させている。

【0009】

【発明が解決しようとする課題】ところが、従来の暗号ソフトの処理方法には次のような問題点がある。すなわち、ユーザーがマウスを用いて所定のファイルを非暗号領域から暗号領域へファイル移動の操作（ドラッグ&ドロップ操作）を行うと、OSは移動要求を発生するので、暗号ソフトはこれを検知し移動完了応答をOSに返送しているようにしている。しかし、実際は図12で示す一連の処理が続行しており、これらの処理が終了しないと、実施にはファイルの移動は完了しない。

【0010】よって、移動するファイルが一つの場合には、移動完了応答を返送後図12の処理は殆ど瞬時に処理されるので問題ないが、移動する対象として複数のファイルを格納したフォルダを指定したような場合、図12の処理に時間がかかってしまう。従って、ユーザーは図12の処理中であるにも係らず処理が終わったものと勘違いし他の作業を進めてしまう。その結果、ユーザーは図12の処理中であるにもかかわらず、前記移動完了応答によって画面に更新表示されたファイルをマウスで操作し、誤って移動先のファイルのどれかを他の場所へ移動させてしまうといったことを実行してしまう危険

性が生じてしまう。このような誤操作をした場合、ユーザーは移動先にファイルが正しく移動したものだと思いこんでいるにも係わらず、実際移動先にファイルは正しく移動せずに、ハードディスク内のどこかに削除されずに残ってしまうという現象が発生してしまう。

【0011】さらに、ユーザーが図12の処理中に処理が終わったものと勘違いして、席を離れてしまうといったことも考えられる。このとき、第3者がユーザーのパソコンを直接操作して暗号化する前の移動元ファイルを取得したり、或いはユーザーのパソコンにネットワークを介して侵入し暗号化処理中の移動元ファイルを不正に取得してしまうといった隙が発生し易い。

【0012】本発明は上記問題を解決する為になされたものであって、暗号化/復号を伴うファイル移動処理中にマウス誤操作によって発生する現象を防止するとともに、暗号処理中のデータを第3者によって不正に取得されるのを防止できる暗号ソフトのデータ移動及び複製方法を提供することを目的とする。

【0013】

【課題を解決しようとする手段】上記目的を解決するために、本発明に係わる電子ファイルのデータ移動、複製方法及び暗号化、復号方法の請求項1記載の発明は、コンピュータシステムの特定のフォルダを暗号領域と設定し、該暗号領域内に格納された暗号化ファイルを、復号した上で前記フォルダ以外の非暗号領域に移動又は複製する方法であって、ハードディスク等のデバイスドライバをコントロールする為の制御関数を取得するステップと、暗号領域内の暗号化ファイルが非暗号領域への移動又は複製が実行されることを検知すると前記制御関数を用いて当該暗号化ファイルをオープンするステップと、オープンした暗号化ファイルを前記制御関数を用いて読み出し、これを復号して平文ファイルを生成するステップと、移動先又は複製先にて前記制御関数を用いてファイルをオープンするステップと、前記平文ファイルを前記制御関数を用いて移動先又は複製先に書き込むステップと、移動先又は複製先の前記平文ファイルを前記制御関数を用いてクローズするステップと、移動の場合、暗号領域内の前記暗号化ファイルを前記制御関数を用いて削除するステップとを備えたものである。

【0014】本発明に係る電子ファイルのデータ移動、複製方法及び暗号化、復号方法の請求項2記載の発明は、コンピュータシステムの特定のフォルダを暗号領域と設定し、前記フォルダ以外の非暗号領域内に格納された平文ファイルを、暗号化した上で前記暗号領域に移動又は複製する方法であって、ハードディスク等のデバイスドライバをコントロールする為の制御関数を取得するステップと、非暗号領域内の平文ファイルが暗号領域への移動又は複製が実行されることを検知すると前記制御関数を用いて当該平文ファイルをオープンするステップと、オープンした平文ファイルを前記制御関数を用いて

読み出し、これを暗号化して暗号化ファイルを生成するステップと、移動先又は複製先にて前記制御関数を用いてファイルをオープンするステップと、前記暗号化ファイルを前記制御関数を用いて移動先又は複製先に書き込むステップと、移動先又は複製先の前記暗号化ファイルを前記制御関数を用いてクローズするステップと、移動の場合、非暗号領域内の前記平文ファイルを前記制御関数を用いて削除するステップとを備えたものである。

【0015】本発明に係る電子ファイルのデータ移動方法及び複製方法の請求項3記載の発明は、請求項1又は請求項2において前記移動する対象がフォルダであった場合、当該フォルダ内のデータを順次請求項1又は請求項2に記載の方法にて移動する第1のステップと、前記データがサブフォルダであった場合、該サブフォルダより上位のフォルダ内の他のデータに先立って当該サブフォルダ内のデータを順次請求項1又は請求項2記載の方法にて移動する第2のステップと、前記第2のステップを繰り返し、移動すべきデータが尽きると移動対象の最上位フォルダに戻って、前記第1のステップと第2のステップとを移動対象のフォルダ内のデータが全て移動するまで実行したものである。

【0016】本発明に係る電子ファイルのデータ移動方法及び複製方法の請求項4記載の発明は、コンピュータシステムにてデータを暗号化する、若しくは暗号化されたデータを復号する方法であって、事前に使用者を設定するために使用者に識別符号の入力を促すステップと、入力された識別符号のメッセージダイジェストを生成するステップと、生成したメッセージダイジェストを複数のデータ列に分割するステップと、分割した前記データ列の少なくとも一つを当該使用者の認証データとして記憶するステップとを備え、所要のデータに対して暗号化又は復号を施すために使用者に識別符号のメッセージダイジェストを生成するステップと、生成されたメッセージダイジェストを複数のデータ列に分割するステップと、分割した前記データ列から前記認証データに相当するものを抽出するステップと、これを予め記憶した前記認証データと比較するステップと、認証データが一致したときに前記データ列のうち前記認証データ以外のデータ列の少なくとも一つを暗号鍵として暗号化又は復号を実施するステップとを備えたものである。

【0017】

【発明の実施の形態】以下図示した実施の形態例に基づいて本発明を詳細に説明する。図1は本発明の暗号ソフトをパーソナルコンピュータシステムにインストールした状況を示したブロック図である。図1において、パーソナルコンピュータシステムにはファイルシステム1を有するOS（オペレーションシステム）2がインストールされているとともに、ハードディスク3を制御する為のハードディスクドライバ4がインストールされている。暗号ソフト5は主にファイルシステム1とハードデ

ィスクドライバ4間に介在してハードディスク3内の所定の電子ファイルにアクセス出来るようになっており、共に、オペレーションシステム2に対して所定の指示を与えるようになっている。なお、パーソナルコンピュータシステムとしてはこの他にマウス、キーボード、CRT等を備えているが説明を簡単にするために図示を省略する。

【0018】以下図1に示した暗号ソフトについてその動作を説明する。まず、暗号ソフトが起動した状態で、ユーザーはCRT等の表示画面を見ながらマウスをドラッグ&ドロップ操作し、暗号領域として指定した所定のフォルダ内に所定の平文ファイル（暗号化していない通常の電子ファイル）を移動させる。前記ファイル移動操作をオペレーティングシステム2は検知し、所定フォルダ（以下移動元フォルダと言う）内の所定ファイルを暗号領域として指定したフォルダ（以下移動先フォルダと言う）内へ移動するように移動要求をファイルシステム1に送出し、ファイルシステム1はこれを暗号ソフト5に送出する。

【0019】次に、暗号ソフト5はファイルシステム1から移動要求を受信すると、移動要求に含まれる情報を元に、以下図2に示したSTEP1～STEP10を順次実行するように、ハードディスクドライバ4（ハードディスク3）に指示する。すなわち、暗号ソフト5はまず、前記移動要求に含まれる情報から移動元ファイルの属性（移動元ファイルのパス、日付等の情報）を読み出すように指示する（STEP1）。そして、入手した移動元ファイルの属性を基に、移動元ファイルをオープン（開く）と共に移動先ファイルをオープン（開く）ように指示する（STEP2、3）。更に、暗号ソフト5は移動元ファイルをハードディスク3から読み込むようにハードディスクドライバ4に指示する（STEP4）。次に、暗号ソフト5はハードディスク3から読み込んだ移動元ファイルを暗号化し（STEP5）、これをハードディスク3内に移動先ファイルとして書き込むようハードディスクドライバ4に指示する（STEP6）。ハードディスク3への書き込みが終了すると、暗号ソフト5は開いていた移動元ファイル及び移動先ファイルをクローズ（閉じる）様にハードディスクドライバ4にそれぞれ指示する（STEP7、8）。

【0020】更に、暗号ソフト5はハードディスク3に保存した移動先のファイル属性でファイル日付を修正するようにハードディスクドライバ4に指示する（STEP9）。これは、移動元ファイルを暗号化し移動先ファイルに書き込んだ時点で、ファイルの日付が更新されてしまうのを防止する目的で実行している。そして、暗号ソフト5は移動元ファイルを削除する様にハードディスクドライバ4に指示を出し（STEP10）、最終的に移動元ファイルはハードディスク3から削除される。

【0021】このように、図2に示したSTEP1～1

0の一連の処理によって、移動元ファイルは自動的に暗号化された上で移動先ファイルに複製され、更に移動元ファイルは削除されるので、あたかもファイルが暗号化されて移動しているように処理されたことになる。また、暗号領域から非暗号領域へファイルを移動させる場合も同様に、移動元ファイルは自動的に復号されて移動しているように処理される。そして、暗号ソフト5は全ての処理が終了すると移動完了応答をOS（ファイルシステム1）に返送する。

【0022】ここで、暗号ソフト5はSTEP5の暗号化／復号する処理以外は全てハードディスクドライバ4に指示を与えて処理を実行させている。しかしながら、暗号ソフト5がOS或いはファイルシステム1を介さずにハードディスクドライバ4に直接アクセスしSTEP1～10を実行できるようにするには、暗号ソフト5が予めハードディスクドライバ4への制御関数を取得しておかなければならない。以下、図2の各STEPを実行するために必要な制御関数を取得する手順を説明する。

【0023】まず、図2のSTEP1実行するために必要な手順について説明する。図3は移動元ファイルの属性を読み出す為の手順を示したものである。暗号ソフト5には事前に適当な名前を付与した架空ファイル（ここではファイル名'X'とする）を設定しておく。暗号ソフト5は架空ファイルの属性を読み出すようOS及びファイルシステム1を介してハードディスクドライバ4（ハードディスク3）に指示する。

【0024】ここで、前記架空ファイルはハードディスク3上には実際存在しないものなので、属性を読み出す処理はエラーとなり処理は終了する。しかしながらこの属性を読み出す処理を1回でも実行すればファイルシステム1からハードディスクドライバ4へ属性の読み出しと付与を実行する為の制御関数が送出されるので、これを暗号ソフト5にて取得し保持しておく。以後、ファイルシステム1を介さずに暗号ソフト5から直接ハードディスクドライバ4へ制御関数で指示することができるようになる。なお、前記架空ファイルと同じファイル名のものが偶然にもハードディスク上に存在したとしても、属性の読み出しが成功するだけで、制御関数が送出されることには変わらない。よって、以上説明した手順で移動元の属性の読み出しと付与を行う制御関数を取得することができる。

【0025】次に、図2のSTEP2を実行するために必要な、ファイルをオープンする制御関数を取得する手順について説明する。図4はファイルオープンする制御関数を取得する手順を示したものである。暗号ソフト5には事前に適当な名前を付与した架空ファイル（ここではファイル名'X'とする）を設定してある。暗号ソフト5はまず、OS及びファイルシステム1を介してハードディスクドライバ4（ハードディスク3）にアクセスし、ドライブのルートを指定してその中の要素（ファイ

ル或いはフォルダ)を探す。そしてファイルが一つも見つからない場合前記架空ファイルをオープン(開く)ようOSに指示する。このとき、前記架空ファイルを読みとり専用モードで開くと、架空ファイルは作成されなくなるのでメモリを消費しないので好ましい。

【0026】ここで、前記架空ファイルはハードディスク3上には実際存在しないものなので、ファイルをオープンする処理はエラーとなり処理は終了する。しかしながらこのファイルをオープンする処理を1回でも実行すればファイルシステムからハードディスクドライバ4へファイルをオープンするための制御関数が出送される。これを暗号ソフト5で取得し保持しておく。以後、ファイルシステム1を介さずに暗号ソフト5から直接ハードディスクドライバ4へ制御関数で指示することができるようになる。

【0027】また、ドライブのルートを指定してその中の要素(ファイルやフォルダ)を探した結果、適当なフォルダ若しくはファイルが見つかった場合は、見つかったフォルダをファイルと見なしてオープンする様にOSに指示し、或いは見つかったファイルをフォルダと見なしてそのフォルダ内の架空ファイルX(実際には存在しないファイル)をオープンするようにOSに指示する。このようにすると、いずれの場合でも処理結果がエラーとなるが、同様にファイルシステム1からハードディスクドライバ4へファイルをオープンするための制御関数が出送される。

【0028】なお、上記手順でファイルをオープンする制御関数を取得すると、その過程でファイルを読む制御関数とファイルに書き込む制御関数とファイルをクローズする制御関数を同時に取得できる。よって、ファイルをオープンする制御関数を取得することによって、図2のSTEP4、6、7、8を実行するために必要な制御関数も取得することができる。

【0029】次に、図2のSTEP10を実行するために必要な、ファイルを削除する制御関数を取得する手順を説明する。図5はファイルを削除する制御関数を取得する手順を示したものである。暗号ソフト5には事前に適当な名前を付与した架空ファイル(ここではファイル名'X'とする)を設定してある。暗号ソフト5はまず、OS及びファイルシステム1を介してハードディスクドライバ4(ハードディスク3)にアクセスし、ドライブのルートを指定してその中の要素(ファイル或いはフォルダ)を探す。そしてファイルが一つも見つからない場合は前記架空ファイルをハードディスク3から削除するようにOSに指示する。このとき、ハードディスク3内に前記架空ファイルは存在しないからエラーとなり処理を終了する。しかしながら、ファイルを削除する処理を1回でも実行すればファイルシステム1からハードディスクドライバ4へファイルを削除するための制御関数が出送される。これを暗号ソフト5にて取得し保持し

ておく。以後、ファイルシステムを介さずに暗号ソフト5から直接ハードディスクドライバ4へ制御関数で指示することができるようになる。

【0030】なお、前記ドライブのルートを指定してその中の要素(ファイル或いはフォルダ)を探した結果、フォルダ若しくはファイルが見つかった場合は、見つかったフォルダをファイルと見なして削除するようOSに指示する。或いは見つかったファイルをフォルダと見なしてそのフォルダ内の架空ファイル'X'を削除するようOSに指示する。いずれの指示でも処理結果がエラーとなり処理を終了するが、同様にファイルシステム1からハードディスクドライバ4へファイルを削除するための制御関数が出送されることになる。以上で、図2のSTEP1~STEP4、STEP6~10を実行するために必要な制御関数を取得することが可能になる。

【0031】更に、ユーザーが移動元の対象として所定のフォルダを選択した場合は、移動対象として前記フォルダは勿論、その中に存在するファイルやサブフォルダ、更には当該サブフォルダ内のファイルをも移動することになる。この場合は、図6にその一例を示すように、いわゆる再帰的手法に基づく手順にて移動処理を行うのが一般的である。この処理方法は移動処理中において、どのファイル又はフォルダを移動したかを常時記憶しておかなければならないためメインメモリを相当量消費してしまう。

【0032】そこで、本発明においてはメインメモリの消費量を抑えたフォルダの移動処理方法(以下非再帰的手法と言う)を提案する。これを図7に具体的な移動元フォルダの構成例を挙げて詳細に説明する。図7において、○印はフォルダを、△印はファイルを示している。今、移動元の最上位フォルダをフォルダ1と仮定しその中にはファイル1とファイル2とサブフォルダ(フォルダ2、5)が存在している。更に、フォルダ2の下にはサブフォルダやファイルが複数存在し、全体として5つのフォルダ(フォルダ1~フォルダ5)と8つのファイル(ファイル1~8)が存在している。

【0033】まず、暗号ソフトは移動元フォルダの最上位フォルダ(フォルダ1)の属性を読み出すようにハードディスクドライバ4に指示する。そして、前記読み出した属性の情報を基に、移動先にフォルダ1の複製を作成し移動先フォルダ(移動先フォルダ1)とする。さらに移動先フォルダ1に属性を付与(日付情報の修正)すると、フォルダ1に戻りフォルダ1内の最初の要素(ファイルやサブフォルダの総称をいう)を探す。フォルダ1内には、ファイル1、2及びフォルダ2、5が存在するが、まずファイル1、及びファイル2をそれぞれ図2に示した手順で暗号化若しくは復号しながら移動先フォルダ1内に移動させる。このとき移動処理を終えたファイル1及びファイル2は移動元フォルダであるフォルダ1から削除された状態になっている。

10

20

30

40

50

【0034】この状態ではフォルダ1の直下にはファイルが一つも存在していない。そこで、ファイルが一つも存在しない状態となると。その下のフォルダ2に移り、これを新たな移動元フォルダとしてその属性を読み出す。前記読み出したフォルダ2の属性を基に、同様に移動先にフォルダ2の複製を作成し移動先フォルダ（移動先フォルダ2）とする。さらに移動先フォルダ2に属性を付与（日付情報の修正）すると、フォルダ2に戻りフォルダ2内の要素を探す。フォルダ2内の要素としてファイル3、4及びフォルダ3、4が見つかるが、ファイルが見つかった場合は、同様に図2に示す手順で暗号化若しくは復号しながらファイル3、及びファイル4を移動先フォルダ2内へ移動させる。このとき移動処理を終えたファイル3及びファイル4は移動元フォルダであるフォルダ2から削除された状態になっている。また、フォルダ2内の要素としてサブフォルダが見つかる（ここではフォルダ3、4が該当）次にサブフォルダへ移り以下同様な手順を繰り返す。

【0035】ここでは、まずフォルダ3に移り、これを新たな移動元ファイルとしてその属性を読み出す。更に前記読み出したフォルダ3の属性を基に、移動先にフォルダ3の複製を作成し移動先フォルダ（移動先フォルダ3）とする。そして、移動先フォルダ3に属性を付与（日付情報の修正）すると、フォルダ3に戻りフォルダ3内の要素を探す。フォルダ3内には、ファイル5及びファイル6のみが存在する。ファイルが存在した場合は、同様に図2に示す手順で暗号化若しくは復号しながらファイル5及びファイル6を移動先フォルダ3へ移動させる。このとき移動処理を終えたファイル5及びファイル6は移動元フォルダであるフォルダ3から削除された状態となっている。更に、フォルダ3内からファイル5、6を移動後フォルダ3内は要素としてなにも存在しない状態となるので、移動元から空となったフォルダ3を削除する。

【0036】次に、削除したフォルダが最上位フォルダ（フォルダ1）に該当しない場合は、一旦最上位のフォルダ（フォルダ1）へ戻り、最上位のフォルダ1から再び上記手順を繰り返す。以下その手順を説明する。今、移動元にはフォルダ1、フォルダ2、フォルダ4、フォルダ5及びファイル7、8が存在している状態である。その他のファイル及びフォルダは既に移動元から削除されている。

【0037】まず、フォルダ1内の要素を探すと、サブフォルダとしてフォルダ2が見つかる。フォルダ2を新たな移動元とし、フォルダ2の属性を読み出す。以下移動先にフォルダ2の複製を作成するが既に作成済みなので。移動先のフォルダ2内の最初の要素を探す。フォルダ2内にはフォルダ4のみが存在する。そこで探す処理を終了しフォルダ4を新たな移動元とする。

【0038】次に、フォルダ4の属性を読みだし、更に

前記読み出したフォルダ4の属性を基に、移動先にフォルダ4の複製を作成し移動先フォルダ4とする。そして、移動先フォルダ4に属性を付与（日付情報の修正）すると、フォルダ4に戻りフォルダ4内の要素を探す。フォルダ4内には、ファイル7、8のみが存在する。ファイルが存在した場合は、同様に図2に示す手順で暗号化若しくは復号しながらファイル7及びファイル8を移動先フォルダ4へ移動させる。このとき移動処理を終えたファイル7及びファイル8は移動元フォルダであるフォルダ4から削除された状態となっている。更に、フォルダ4内からファイル7、8を移動後フォルダ4内は要素としてなにも存在しない状態となるので、移動元から空となったフォルダ4を削除する。

【0039】次に、削除したフォルダが最上位フォルダ（フォルダ1）に該当しない場合は、最上位のフォルダ（フォルダ1）へ戻り、最上位のフォルダ1から再び同様な手順を繰り返す。以下その手順を説明する。

【0040】今、移動元にはフォルダ2及びフォルダ5が存在している状態である。その他のファイル及びフォルダは既に移動元から削除されている。以下手順に従っていくと、フォルダ2が移動元フォルダとなるが、フォルダ2内は空となっているので削除され最上位のフォルダ（フォルダ1）に戻る。次にフォルダ1内にはサブフォルダとしてフォルダ5が存在するのでこれが移動元フォルダとなる。移動先にフォルダ5の複製を作成し、フォルダ5内は空であるので移動元からフォルダ5を削除する。よって、以上説明した手順で移動元のフォルダ1内の全ファイル及びサブフォルダは移動先に移動した。最後に最上位フォルダである移動元のフォルダ1を削除して移動が完了する。以上、図7を使って説明した処理手順を一般化すると図8に示すフローチャートのとおりとなる。

【0041】次に、図7、8に示したしたフォルダを移動するための各手順を実行するために必要な、制御関数を取得する手順について説明する。図9は移動元フォルダの最初の要素を探す制御関数を取得する手順を示したものである。暗号ソフト5はドライブのルートを指定してその中の最初の要素（フォルダやファイル）を探すようにOSに指示する。その結果要素が見つかったり或いは見つからなくとも、上記処理をOSに実行させた結果、ファイルシステム1からハードディスクドライバ4へ最初の要素を探すための制御関数が送出される。これを暗号ソフト5にて取得し保持する。以後、ファイルシステム1を介さずに暗号ソフト5から直接ハードディスクドライバ4へ制御関数で指示することが出来るようになる。なお、前記最初の要素を探す制御関数を取得した結果、次の要素を探す制御関数を得ることができる。

【0042】次に、移動元フォルダを作成或いは削除する制御関数を取得する手順について説明する。図10は移動元フォルダを作成或いは削除する制御関数を取得す

る手順を示したものである。暗号ソフト5には事前に適
当な名前を付与した架空ファイル（ここではファイル
名'X'とする）を設定してある。暗号ソフト5はま
ず、OS及びファイルシステム1を介してハードディス
クドライバ4（ハードディスク3）にアクセスし、ドラ
イブのルートを指定してその中の要素（ファイル或いは
フォルダ）を探す。

【0043】そしてファイルもフォルダも見つからな
かった場合にはハードディスク3から前記架空ファイル'
X'を削除するようにOSに指示する。また、ファイル
或いはフォルダが見つかった場合、その見つかったファ
イル或いはフォルダと同じ名前のフォルダを作成する
ようにOSに指示する。その結果同名のファイル若しくは
フォルダが既に存在することから、同名でフォルダを作
成することが不可となり処理はエラーとなって終了す
る。しかしながら上記処理をOSに実行させることでフ
ォルダを作成或いは削除する制御関数を取得すること
ができる。

【0044】以上説明した手順によって、暗号ソフト5
は起動時にファイルシステム1を介して必要な制御関数
を取得し、これを用いてハードディスクドライバ4を直
接制御することが可能になるので、移動元ファイル或
いはフォルダを暗号化／復号を行いながら移動先へ移動
させる処理を実施することが可能になる。しかも、暗号
ソフト5からハードディスクドライバ4に制御関数を用
いて直接指示するようにしたことで、全ての処理が終了
した時点でファイルシステム1に移動要求応答を返送で
きるようにした。よって、処理中はファイルにアクセ
スすることができなくなり処理中のファイルを間違って
別の場所に移動させてしまうといったことも防止でき
る。従って、処理中であるにもかかわらず、ユーザ
ーが処理を終わったものと勘違いすることもないし、
処理中にユーザーが離席しても第3者がこれを不正に
取得するといったことも防止できる。更に、第3者が
ユーザーのパソコンにネットワークを介して侵入し、
暗号化処理中のファイルを不正に取得するといったこ
とも防止できる。以上、本発明をファイル等の移動
を例に説明したが複製にも適用可能である。

【0045】次に、暗号ソフトを起動した際、暗号化
或いは復号に必要な鍵データを生成する方法について
説明する。図11は一般的な暗号鍵又は復号鍵を生成
する過程を示したブロック図である。図11に示すよ
うに、暗号ソフトを起動させると、まずモニター上に
ID（識別符号）の入力を促す表示がなされ、ユーザ
ーはこれに従いキーボードからIDを入力する。これ
を基にハッシュ関数演算を実行しハッシュ値と呼ば
れる所定ビットの符号を生成し、更にハッシュ値を
基に所定の暗号鍵を生成する。生成された暗号鍵は
ハードディスクに保存され、所定の電子ファイルを
暗号化若しくは復号する際に使用される。

【0046】ところが、この暗号鍵生成方法では暗号
鍵がハードディスクに保存されているため、ユーザ
ーのコンピュータにネットワークを介して第3者が
侵入し、ハードディスクから暗号鍵を不正に取得し
やすいという欠点を有している。そこで、この欠点
を解消するため、図12に新たな暗号鍵の生成方法
を示す。図12において、ユーザーはモニター上の
ID入力を促す表示に従って、キーボードからIDを
入力する。これを基にハッシュ関数演算を実行して
所定ビットのハッシュ値（メッセージダイジェスト）
を生成する。次に、前記ハッシュ値を二つに分割し、
一方をハッシュ値A、他方をハッシュ値Bとする。
ここで、前記ハッシュ値を複数のデータ列に分割し、
前記複数のデータ列の任意の二つを選択し前記ハ
ッシュ値A及びハッシュ値Bとしてもよい。

【0047】そこで、前記ハッシュ値Aを基に暗号
鍵を生成しこれをメインメモリにのみ保持させてお
き、ソフトを終了させたときにはメインメモリから
暗号鍵を削除するようにしておく。一方、前記ハッ
シュ値Bは認証データとしてハードディスクに保存
する。そして、次回、ユーザーがコンピュータから
IDを入力したときに生成される認証データと前記
保存した認証データとを比較し、認証データが一
致すれば前記ハッシュ値Aを基に暗号鍵を生成し、
所望の暗号化又は復号を実施できるようにする。
認証データが一致しない場合は、生成した暗号
鍵を破棄して暗号化又は復号処理を実行できない
ようにする。

【0048】このようにすると、ハッシュ値Aから
生成された暗号鍵はハードディスク上に保存され
ないので、これを第3者がネットワークを介して
不正に入手するといったことが不可能になる。し
かも、ユーザー本人しか暗号ソフトを起動させる
ことができないため、第3者がユーザーのコンピ
ュータ上で暗号化ファイルを不正に復号するとい
ったことも不可能となり、セキュリティを確保
する上で有効な手段となる。

【0049】

【発明の効果】本発明は以上説明したように、従
来暗号ソフトが暗号化若しくは復号をともなうフ
ァイル移動処理を実行する際、OSを介してハー
ドディスクドライバ（ハードディスク）を制御し
なければならない為、ファイルの移動が完了し
たことを示す移動完了応答をOSに返送した後に
前記暗号化若しくは復号を伴う移動処理を実
施していたときの問題点と、生成した暗号鍵を
第3者から不正に取得されやすい問題点を解決
し、暗号ソフトがハードディスクドライバ（ハー
ドディスク）を直接コントロールする制御関
数を取得する手順を備え、前記所得した制御関
数を用いて暗号化若しくは復号を伴うファイル
移動処理を実行すると共に、生成した暗号鍵を
メインメモリ上にのみ保持しハードディスク上
に保存しないようにしたので、処理安定度が
高く、且つセキュリティ機能の高い暗号ソフト
のデータの移動、複製方法及び暗

号化、復号方法を供給するのに著効を奏す。

【0050】

【図面の簡単な説明】

【図1】本発明に係わる暗号ソフトの構成を示すブロック図。

【図2】本発明に係わる暗号ソフトのファイル移動処理を示したフローチャート図。

【図3】本発明に係わる暗号ソフトの属性読み出し及び属性付与する制御関数を取得する手順を示したフローチャート図。

【図4】本発明に係わる暗号ソフトのファイルをオープンする制御関数を取得する手順を示したフローチャート図。

【図5】本発明に係わる暗号ソフトのファイルを削除する制御関数を取得する手順を示したフローチャート図。

【図6】本発明に係わる暗号ソフトのフォルダ移動処理を示した再帰的手法によるフローチャート図。

【図7】本発明に係わる暗号ソフトの移動元フォルダの一例を示した図。

【図8】本発明に係わる暗号ソフトのフォルダ移動処理*20

*を示した非再帰的手法によるフローチャート図。

【図9】本発明に係わる暗号ソフトの最初の要素を探す制御関数を取得する手順を示したフローチャート図。

【図10】本発明に係わる暗号ソフトのフォルダを作成、削除する制御関数を取得する手順を示したフローチャート図。

【図11】本発明に係わる暗号ソフトの暗号鍵を生成する第1の方法を示すフローチャート図。

【図12】本発明に係わる暗号ソフトの暗号鍵を生成する第2の方法を示すフローチャート図。

【図13】従来の暗号ソフトの構成を示すブロック図。

【図14】従来の暗号ソフトのファイル移動処理を示したフローチャート図。

【符号の説明】

1…ファイルシステム

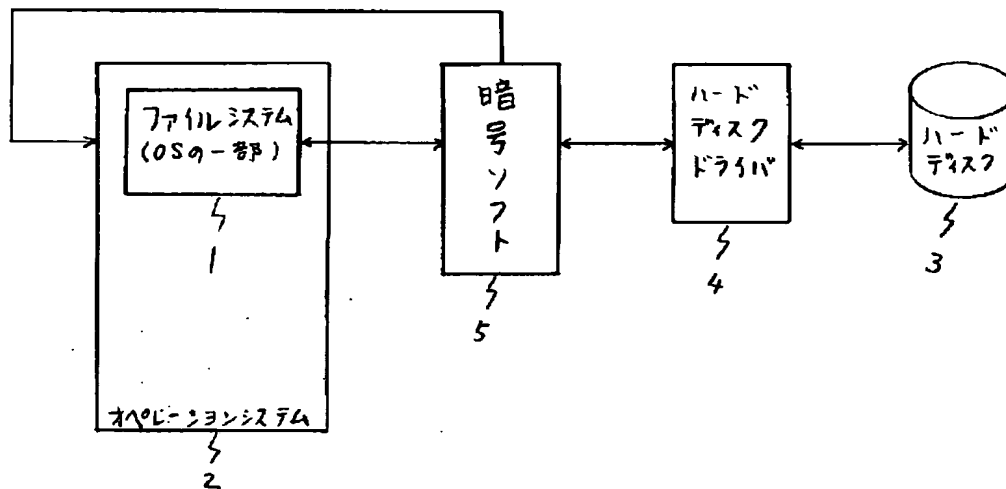
2…オペレーションシステム

3…ハードディスク

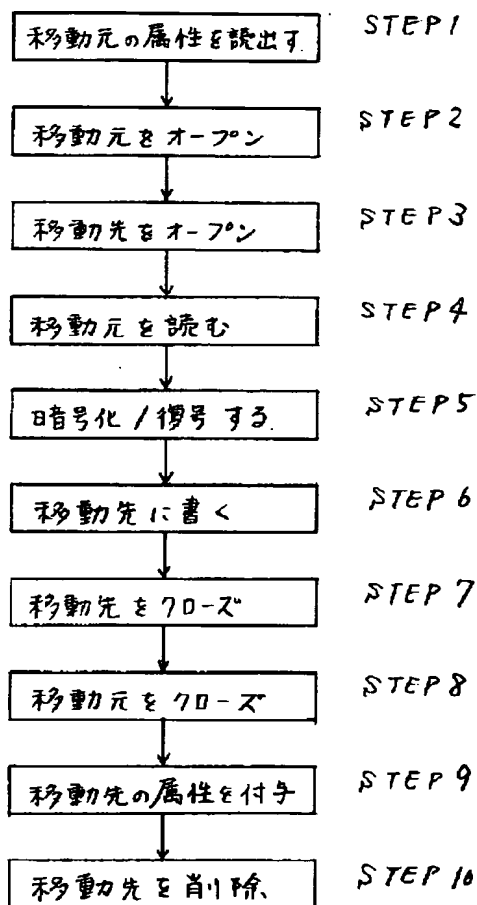
4…ハードディスクドライバ

5…暗号ソフト

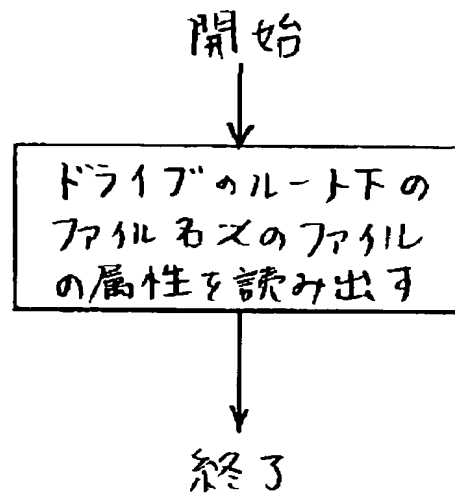
【図1】



【図2】

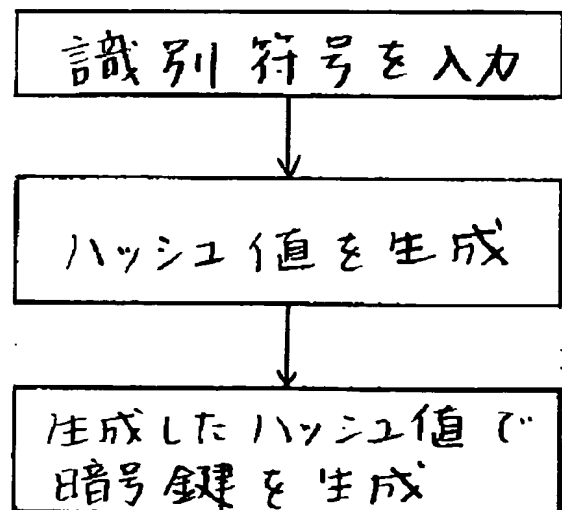


【図3】



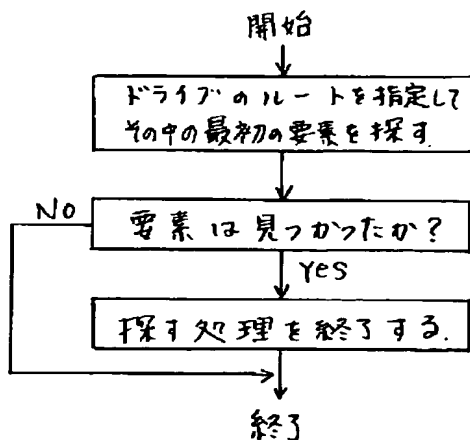
属性読み出し及び属性を付与
する制御関数を取得する手順

【図11】

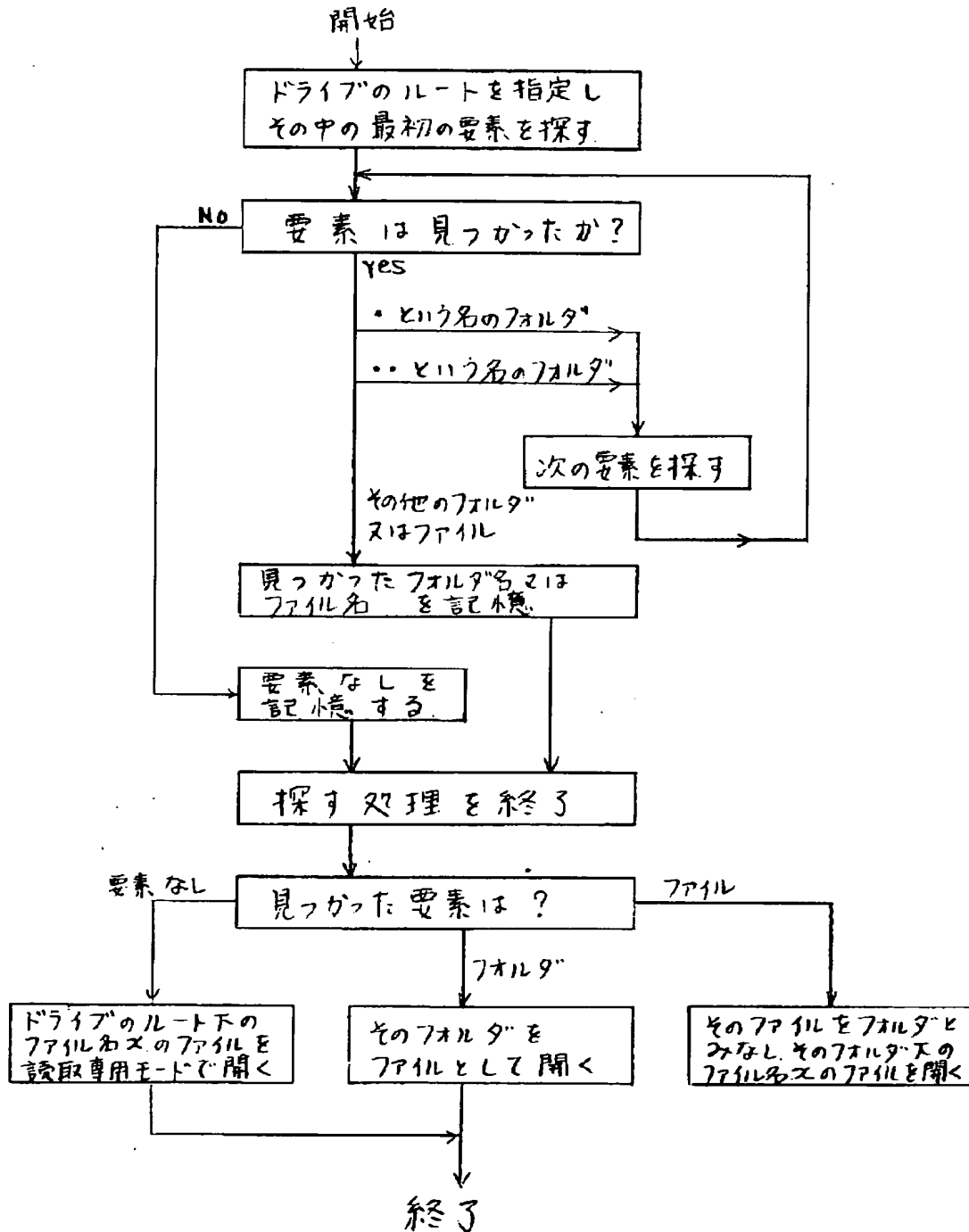


【図9】

最初の要素を探す制御関数を取得する手順

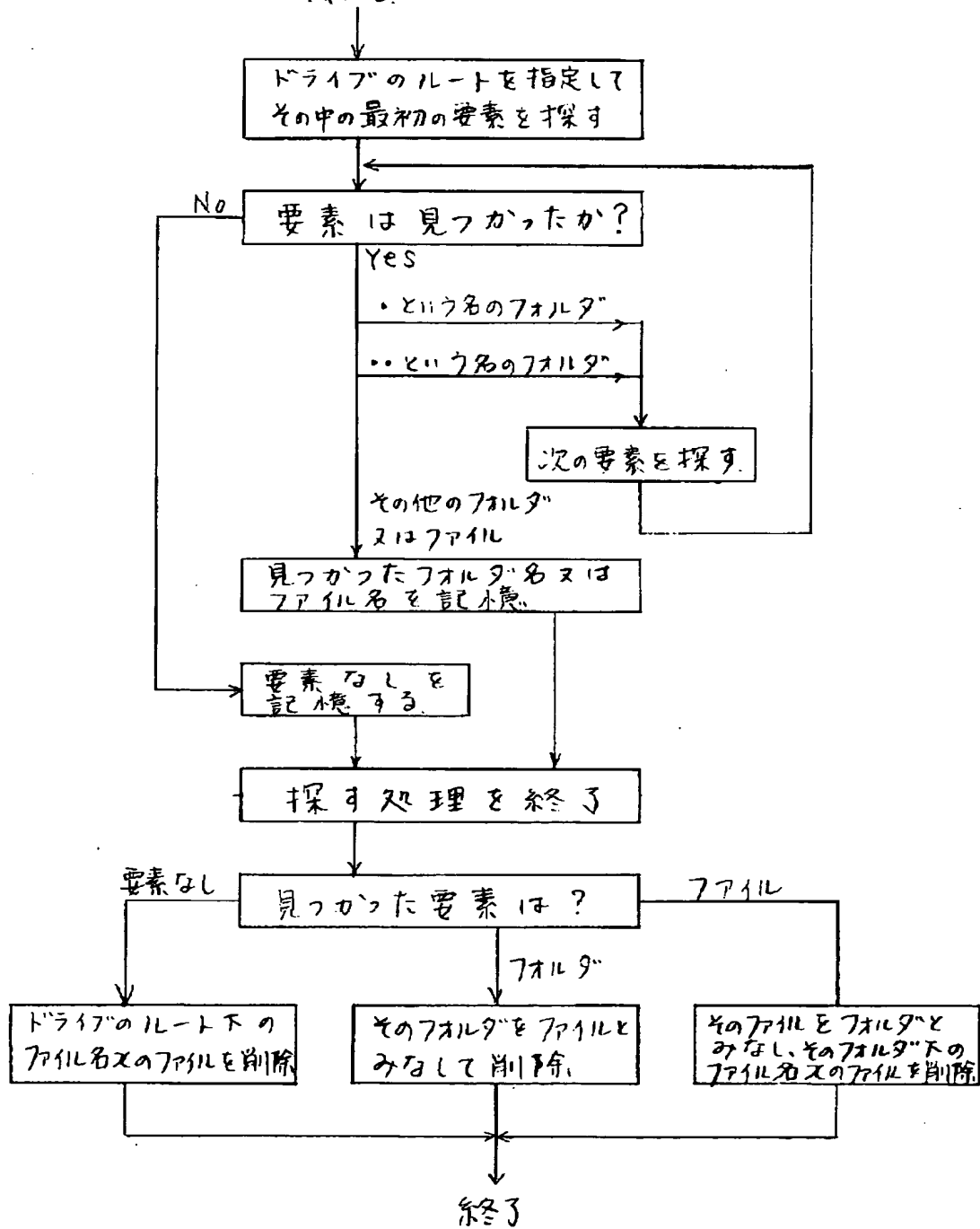


ファイルをオープンする制御関数を取得する手順

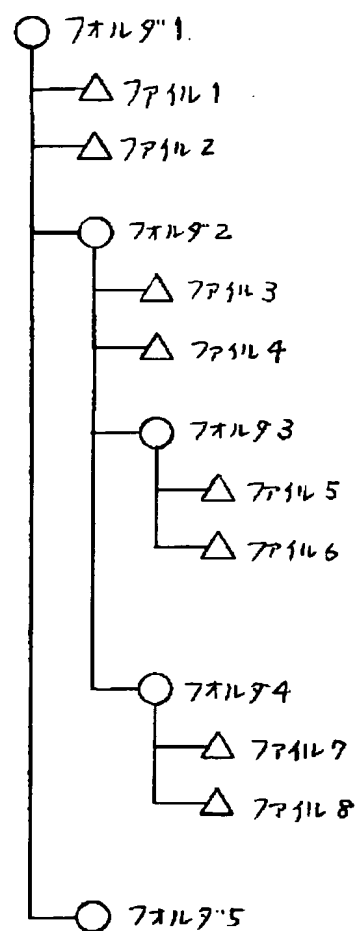


【図5】

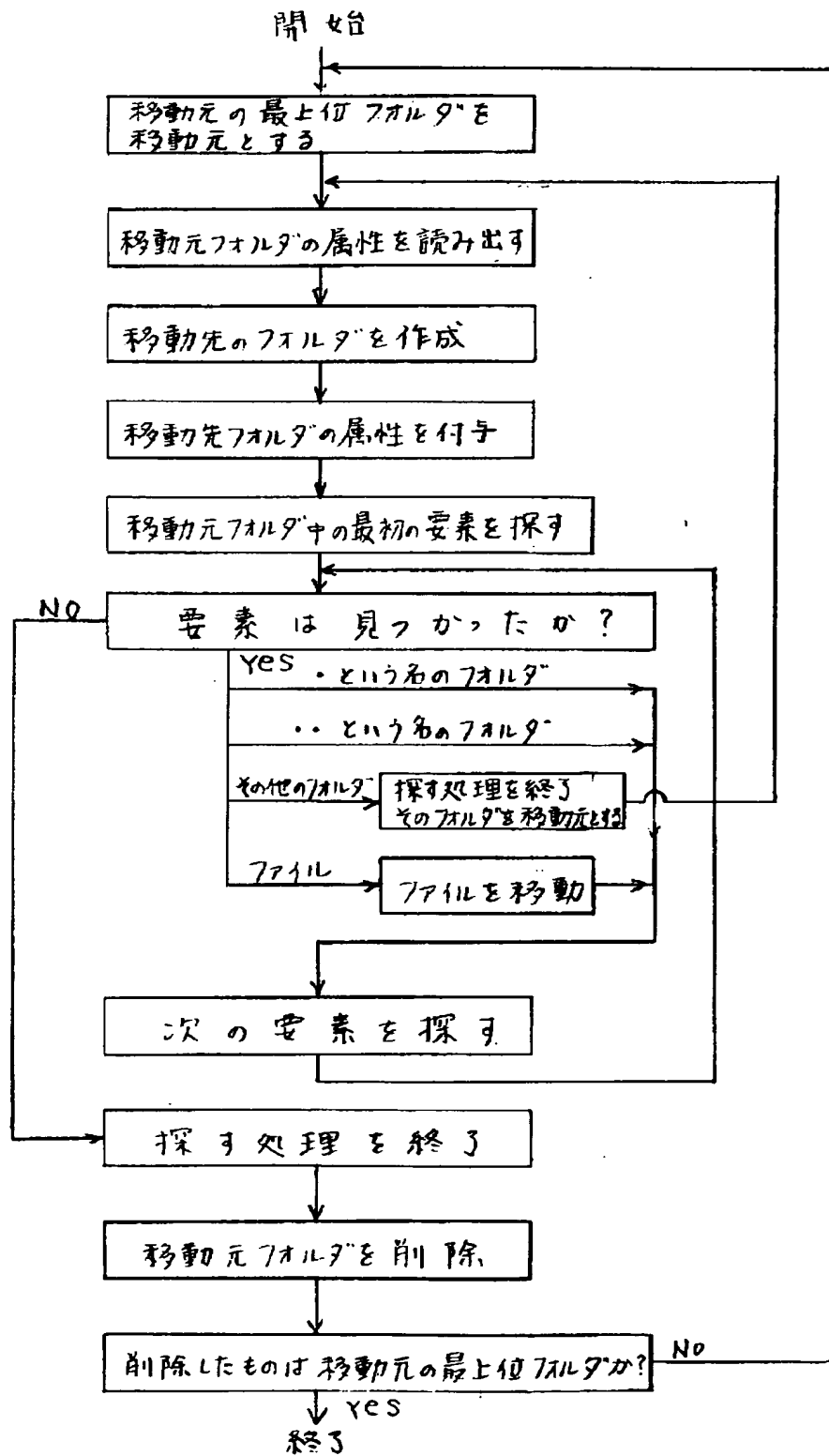
ファイルを削除する制御関数を取得する手順
開始



【图7】

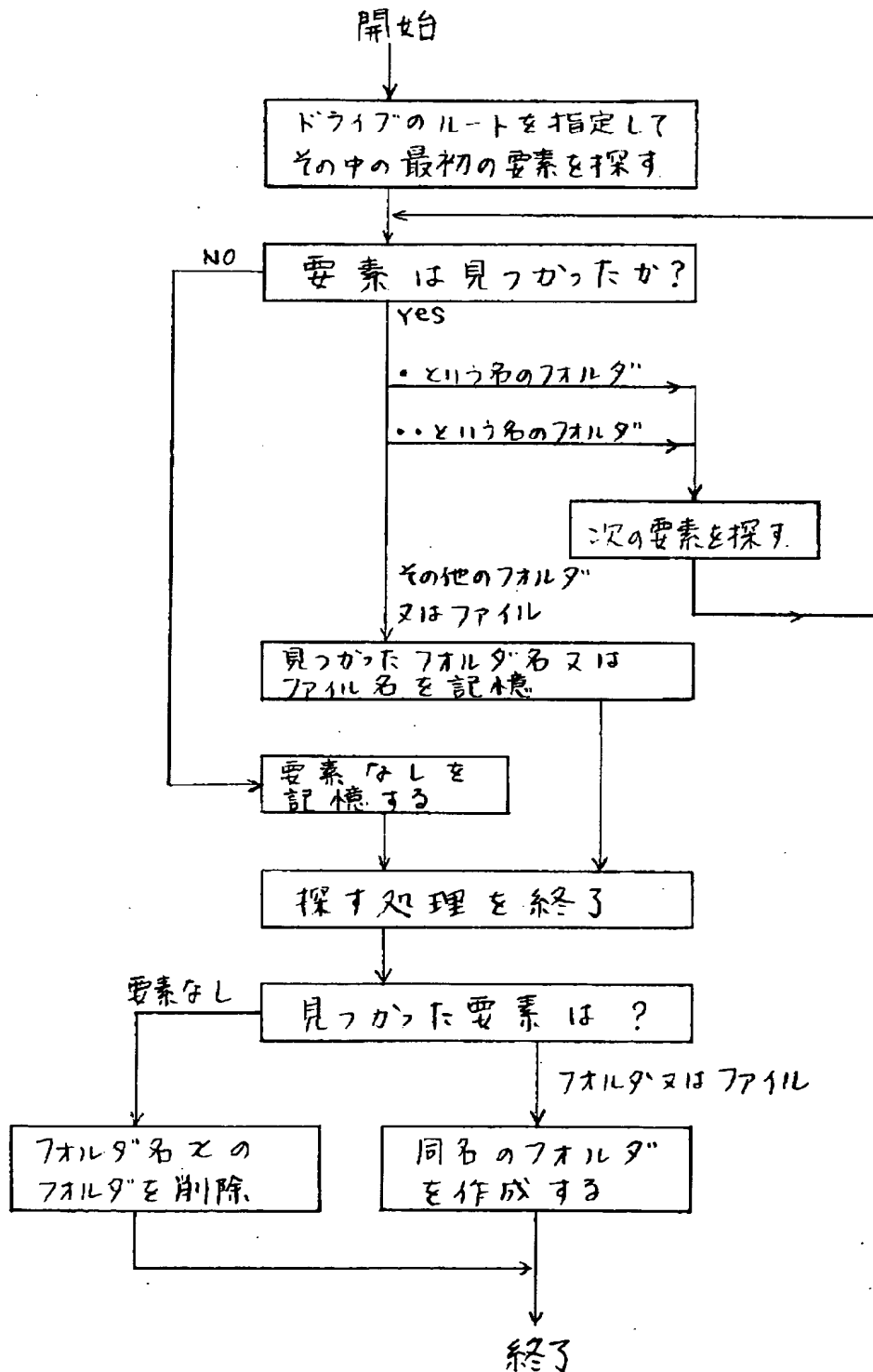


【図8】

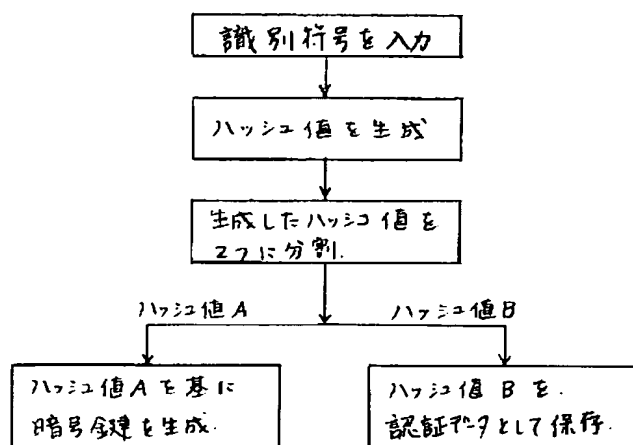


【図10】

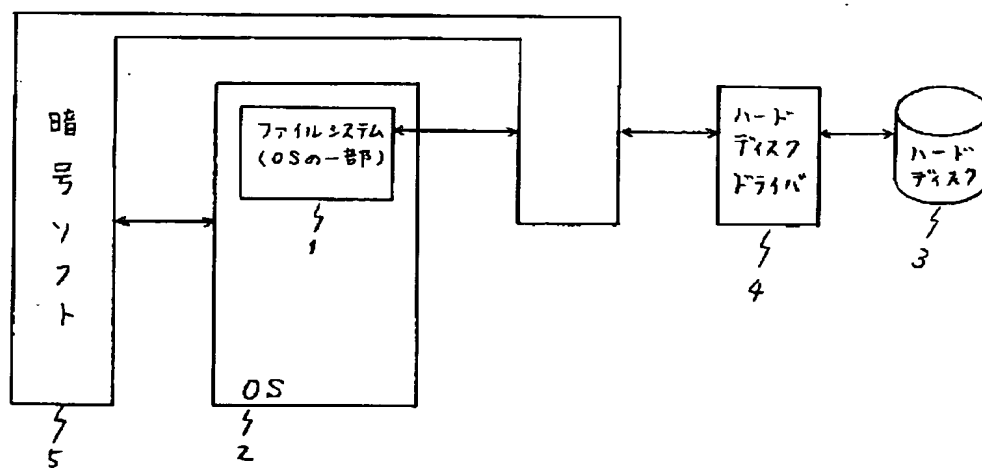
フォルダを作成、削除する制御関数を取得する手順



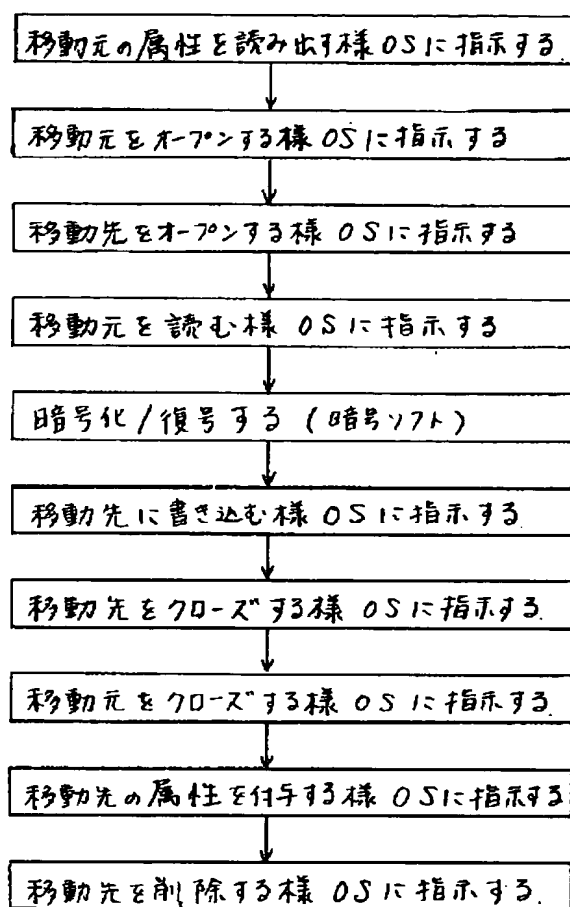
【図12】



【図13】



【図14】



PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-175210

(43)Date of publication of application : 21.06.2002

(51)Int.Cl.

G06F 12/00

G06F 12/14

G09C 1/00

(21)Application number : 2000-373593

(71)Applicant : TOYO COMMUN EQUIP CO LTD

(22)Date of filing : 07.12.2000

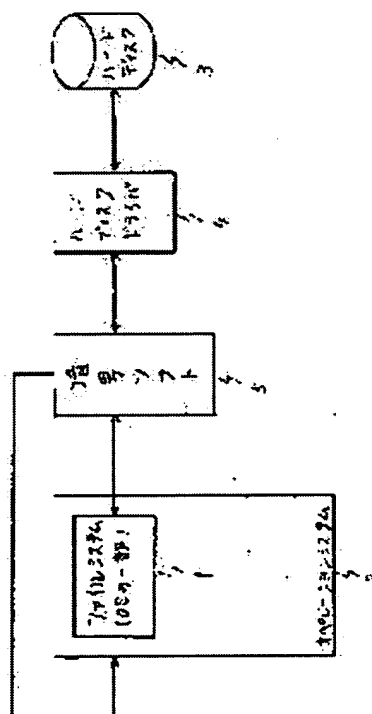
(72)Inventor : ICHISE HIROSHI

(54) METHOD OF TRANSFERRING AND COPYING, AND ENCRYPTING AND DECRYPTING DATA

(57)Abstract:

PROBLEM TO BE SOLVED: To enhance processing stability and a security function.

SOLUTION: This method is provided with a step for obtaining a control function for controlling a device driver such as a hard disk, a step for opening an encryption file using the control function when detecting that the file within a cipher area is transferred or copied to a noncipher area, a step for reading the opened encryption file using the control function to generate an ordinary sentence file by decrypting, a step for opening the file in a transferred site or copied site using the control function, a step for writing the ordinary sentence file in the transferred site or copied site using the control function, a step for closing the ordinary sentence file in the transferred site or copied site using the control function, and a step for deleting the encryption file within the cipher area using the control function in the case of the transfer.



* NOTICES *

JP0 and INPIT are not responsible for any damages caused by the use of this translation.

- 1.This document has been translated by computer. So the translation may not reflect the original precisely.
- 2.**** shows the word which can not be translated.
- 3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]

[Claim 1]How to move or reproduce it to non-code fields other than said folder after decoding an enciphered file which set a specific folder of a computer system characterized by comprising the following to a code field, and was stored in this code field.

A step which acquires a control function for controlling device drivers, such as a hard disk.

A step which opens the enciphered file concerned using said control function if an enciphered file in a code field detects that movement or a duplicate to a non-code field is performed.

A step which reads an opened enciphered file using said control function, decodes this, and generates a plaintext file.

A step which opens a file in a movement destination or a copying destination using said control function, A step which writes said plaintext file in a movement destination or a copying destination using said control function, a step which closes said plaintext file of a movement destination or a copying destination using said control function, and a step which deletes said enciphered file in a code field using said control function in movement.

[Claim 2]How to move or reproduce a plaintext file characterized by comprising the following which set a specific folder of a computer system to a code field, and was stored in non-code fields other than said folder to said code field after enciphering.

A step which acquires a control function for controlling device drivers, such as a hard disk.

A step which opens the plaintext file concerned using said control function if a plaintext file in a non-code field detects that movement or a duplicate to a code field is performed.

A step which reads an opened plaintext file using said control function, enciphers this, and generates an enciphered file.

A step which opens a file in a movement destination or a copying destination using said control function, A step which writes said enciphered file in a movement destination or a copying

destination using said control function, a step which closes said enciphered file of a movement destination or a copying destination using said control function, and a step which deletes said plaintext file in a non-code field using said control function in movement.

[Claim 3]The 1st step that moves data in the folder concerned by a method according to claim 1 or 2 one by one when said object which moves is a folder, The 2nd step that moves data in the subfolder concerned by a method according to claim 1 or 2 one by one in advance of other data in a folder of a higher rank from this subfolder when said data is a subfolder, Movement of data performing until it will return to the top folder of a moved object and all data in a folder of a moved object will move said 1st step and the 2nd step, if data which should repeat said 2nd step and should move is exhausted, a duplicating method and encryption, a decoding method.

[Claim 4]Movement of data characterized by comprising the following, a duplicating method and encryption, a decoding method.

A step which data is enciphered in a computer system, or is how to decode enciphered data, and demands an input of an identification signal from a user in order to set up a user a priori.

A step which generates a message digest of an inputted identification signal.

A step which divides a generated message digest into multiple-data-stream.

A step which generates a message digest of an identification signal to a user in order to have a step which memorizes at least one of said the divided data rows as authentication data of the user concerned and to give encryption or decoding to necessary data, A step which divides a generated message digest into multiple-data-stream, A step which extracts a thing equivalent to said authentication data from said divided data row, A step in comparison with said authentication data which memorized this beforehand, and a step which carries out encryption or decoding by using at least one of said data rows of data rows other than said authentication data as an encryption key when authentication data is in agreement.

[Translation done.]

* NOTICES *

JP0 and INPIT are not responsible for any damages caused by the use of this translation.

- 1.This document has been translated by computer. So the translation may not reflect the original precisely.
- 2.**** shows the word which can not be translated.
- 3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention]This invention relates to the method of moving an electronic file, enciphering or decoding an electronic file in a personal computer system etc.

[0002]

[Description of the Prior Art]In recent years, the encryption software which can encipher easily electronic files created with the personal computer etc., such as image data and text data, is sold. For example, if software is started, a predetermined password is entered first, a user is authorized and the encryption software of the type which specifies a predetermined folder as an enciphering area next is known. If it is going to save electronic files, such as text data created with predetermined application when this software was stationed permanently on the computer, to said code field, encryption software perceives this, enciphers said electronic file, and is saved. Since said electronic file will decode and will pass processing to predetermined applications (for example, word-processing software etc.) if the electronic file in a code field is double-clicked with a mouse, On a screen, an electronic file is displayed as a usual text file.

[0003]On the other hand, in the computer which sets Windows 95 of Microsoft Corp., etc. to OS (operation system), Explorer is preinstalled as a tool for file management, and movement and a copy of a file can be performed easily. However, when the electronic file enciphered using this Explorer is moved to a non-code field from a code field, a file is not automatically decoded only by a pathname (preservation destination folder name) being changed as for said electronic file. Even if it moves the usual electronic file to a code field from a non-code field, it is not automatically enciphered only by the pathname of a file being changed too. Then, encryption software with the function which perceives operation of file migration, and enciphers or decodes an electronic file automatically is proposed.

[0004]Drawing 13 is a block diagram showing briefly the state where the conventional

encryption software was installed in a personal computer system. In drawing 13, while OS (operation system) 2 which has the file system 1 is installed in the personal computer, the hard disk driver 4 for controlling the hard disk 3 is installed. The encryption software 5 can access now the predetermined electronic file in the hard disk 3 via the operation system 2 and the hard disk driver 4. Here, some encryption software 5 intervenes between the operation system 2 and the hard disk driver 4. A graphic display is omitted in order to explain simply, although it has a mouse, a keyboard, CRT, etc. as a personal computer system.

[0005]Drawing 14 is a flow chart figure showing the example of the conventional encryption software shown in drawing 13 of operation. The example of the conventional encryption software of operation is explained below, using drawing 14. First, after encryption software has started, a user does drag-and-drop operation of the mouse, looking at display screens, such as CRT, and moves a predetermined file (the usual electronic file which has not been enciphered) into the predetermined folder specified as a code field. The operating system 2 detects said file migration operation, and a move demand is sent out to the file system 1 so that it may move into the folder (henceforth a movement destination folder) which specified the predetermined file in a predetermined folder (henceforth a source folder) as a code field, The file system 1 sends this out to the encryption software 5.

[0006]Here, the encryption software 5 is monitoring continuously the information included in said move demand. If said move demand directs movement into [out of a non-code field] a code field, or movement into [out of a code field] a non-code field, The encryption software 5 does not hand over this move demand to the hard disk driver 4, Cipher processing which returned and showed the completion response of a move which shows that movement was completed continuously to the operation system 2 by drawing 12 via the file system 1 is performed via the operation system 2. This is because the operation system 2 will receive the next processing if the completion response of a move is not returned to said move demand. Therefore, the encryption software 5 returns the completion response of a move to the operation system 5, and he is trying to direct processing of drawing 12 to the operation system 5 separately to said move demand.

[0007]Then, it is directed to the OPE ration system 2 (henceforth OS) that the encryption software 5 reads the attribute (information, including the path of a move original file, a date, etc.) of a move original file from the information first included in said move demand as shown in drawing 12. And a move original file is directed to OS openly (it opens) like based on the attribute of the move original file which came to hand, and it is directed to OS that a move original file reads from the hard disk 3. Next, the encryption software 5 enciphers the move original file read from the hard disk 3, and it directs to write this in as a movement destination file in the hard disk 3 to OS. After writing is completed, the encryption software 5 points to the open move original file to OS at Mr. Closing (it closes), and directs the movement destination

file which was further open to closing (it closes) Mr. OS.

[0008]It is directed to OS that the encryption software 5 corrects a file date by the file attribute of the movement destination saved at the hard disk 3. When a move original file is enciphered and it writes in a movement destination file, this is performed in order to prevent the date of a file from being updated. And the encryption software 5 takes out directions to OS so that a move original file may be deleted, and a move original file is eventually deleted from the hard disk 3. Thus, since a move original file is reproduced by the movement destination file after being enciphered automatically, and also a move original file is deleted by a series of processings shown in drawing 12, it means that it was processed as a file was enciphered and it was moving. Also when a file is moved to a non-code field from a code field, as it is shown in drawing 12, a move original file is processed as it is decoded automatically and is moving. Thus, except [all] the processing enciphered / decoded, the encryption software 5 gives directions to OS and is performing processing.

[0009]

[Problem(s) to be Solved by the Invention]However, there are the following problems in the disposal method of the conventional encryption software. That is, since OS will generate a move demand if a user performs operation (drag-and-drop operation) of file migration for a predetermined file from a non-code field to a code field using a mouse, encryption software detected this and he is trying to have returned the completion response of a move to OS. However, unless a series of processings shown by drawing 12 in practice continue and these processings are completed, movement of a file is not completed for operation.

[0010]Therefore, when it was satisfactory since most processings of after-return drawing 12 were processed in an instant in the completion response of a move when the file which moves was one, but the folder which stored two or more files is specified as an object which moves, processing of drawing 12 will take time. Therefore, although a user is during processing of drawing 12, he will mistake with what processing finished and will do other work. As a result, although a user is during processing of drawing 12, the danger of performing operating on a screen the file by which it was indicated by updating with a mouse, and making it moving one of the files of a movement destination to other places accidentally by said completion response of a move will produce him. When such an operation mistake is carried out, in spite of having convinced the user that the file moved to the movement destination correctly, the phenomenon of remaining without being deleted by somewhere in a hard disk without a file actually moving to a movement destination correctly will generate him.

[0011]It is also considered that a user will mistake with what processing finished during processing of drawing 12, and will leave a seat. At this time, it is easy to generate the spare time of acquiring the move original file before the 3rd person does the direct control of a user's personal computer and enciphers, or invading into a user's personal computer via a network,

and acquiring the move original file under encryption processing unjustly.

[0012]This invention was made in order to solve the above-mentioned problem, and it is ****. The purpose is providing the data movement and the duplicating method of the encryption software which can prevent the 3rd person acquiring the data under cipher processing unjustly while preventing the phenomenon generated by a mouse operation mistake during the file migration processing accompanied by /decoding.

[0013]

[Means for Solving the Problem]In order to solve the above-mentioned purpose, data movement of an electronic file concerning this invention, a duplicating method and encryption, and the invention of a decoding method according to claim 1, An enciphered file which set a specific folder of a computer system to a code field, and was stored in this code field, A step which acquires a control function for being the method of moving or reproducing to non-code fields other than said folder after decoding, and controlling device drivers, such as a hard disk, A step which opens the enciphered file concerned using said control function if an enciphered file in a code field detects that movement or a duplicate to a non-code field is performed, A step which reads an opened enciphered file using said control function, decodes this, and generates a plaintext file, A step which opens a file in a movement destination or a copying destination using said control function, A step which writes said plaintext file in a movement destination or a copying destination using said control function, It has a step which closes said plaintext file of a movement destination or a copying destination using said control function, and a step which deletes said enciphered file in a code field using said control function in movement.

[0014]Data movement of an electronic file concerning this invention, a duplicating method and encryption, and the invention of a decoding method according to claim 2, A plaintext file which set a specific folder of a computer system to a code field, and was stored in non-code fields other than said folder, A step which acquires a control function for being the method of moving or reproducing to said code field after enciphering, and controlling device drivers, such as a hard disk, A step which opens the plaintext file concerned using said control function if a plaintext file in a non-code field detects that movement or a duplicate to a code field is performed, A step which reads an opened plaintext file using said control function, enciphers this, and generates an enciphered file, A step which opens a file in a movement destination or a copying destination using said control function, A step which writes said enciphered file in a movement destination or a copying destination using said control function, It has a step which closes said enciphered file of a movement destination or a copying destination using said control function, and a step which deletes said plaintext file in a non-code field using said control function in movement.

[0015]The invention of a data movement method of an electronic file concerning this invention, and a duplicating method according to claim 3, The 1st step that moves data in the folder concerned by a method according to claim 1 or 2 one by one when said object which moves is a folder in claim 1 or claim 2, The 2nd step that moves data in the subfolder concerned by a method according to claim 1 or 2 one by one in advance of other data in a folder of a higher rank from this subfolder when said data is a subfolder, It performs until it will return to the top folder of a moved object and all data in a folder of a moved object will move said 1st step and the 2nd step, if data which should repeat said 2nd step and should move is exhausted.

[0016]The invention of a data movement method of an electronic file concerning this invention, and a duplicating method according to claim 4, A step which enciphers data in a computer system, or demands an input of an identification signal from a user in order to be the method of decoding enciphered data and to set up a user a priori, A step which generates a message digest of an inputted identification signal, A step which divides a generated message digest into multiple-data-stream, A step which generates a message digest of an identification signal to a user in order to have a step which memorizes at least one of said the divided data rows as authentication data of the user concerned and to give encryption or decoding to necessary data, A step which divides a generated message digest into multiple-data-stream, A step which extracts a thing equivalent to said authentication data from said divided data row, It has a step in comparison with said authentication data which memorized this beforehand, and a step which carries out encryption or decoding by using at least one of said data rows of data rows other than said authentication data as an encryption key when authentication data is in agreement.

[0017]

[Embodiment of the Invention]Based on the example of an embodiment illustrated below, this invention is explained in detail. Drawing 1 is a block diagram showing the situation where the encryption software of this invention was installed in the personal computer system. In drawing 1, while OS(operation system) 2 which has the file system 1 is installed in the personal computer system, the hard disk driver 4 for controlling the hard disk 3 is installed. The encryption software 5 mainly intervenes between the file system 1 and the hard disk driver 4, and can access the predetermined electronic file in the hard disk 3, and it gives predetermined directions to the operation system 2. A graphic display is omitted in order to explain simply, although it has a mouse, a keyboard, CRT, etc. in addition to this as a personal computer system.

[0018]The operation is explained about the encryption software shown in drawing 1 below. First, after encryption software has started, a user does drag-and-drop operation of the mouse, looking at display screens, such as CRT, and moves a predetermined plaintext file (the usual electronic file which has not been enciphered) into the predetermined folder specified as a

code field. The operating system 2 detects said file migration operation, and a move demand is sent out to the file system 1 so that it may move into the folder (henceforth a movement destination folder) which specified the predetermined file in a predetermined folder (henceforth a source folder) as a code field, The file system 1 sends this out to the encryption software 5. [0019]Next, if a move demand is received from the file system 1, the encryption software 5 is directed to the hard disk driver 4 (hard disk 3) so that sequential execution of the STEP1-STEP10 which were shown in drawing 2 below may be carried out based on the information included in a move demand. That is, it is directed that the encryption software 5 reads the attribute (information, including the path of a move original file, a date, etc.) of a move original file from the information included in said move demand first (STEP1). And a movement destination file is directed for a move original file openly (it opens) like with opening (it opens) based on the attribute of the move original file which came to hand (STEP 2 and 3). It is directed to the hard disk driver 4 that the encryption software 5 reads a move original file from the hard disk 3 (STEP4). Next, the encryption software 5 enciphers the move original file read from the hard disk 3 (STEP5), and it directs to write this in as a movement destination file in the hard disk 3 to the hard disk driver 4 (STEP6). After the writing to the hard disk 3 is completed, the encryption software 5 directs the open move original file and movement destination file to Mr. Closing (it closes) at the hard disk driver 4, respectively (STEP 7 and 8). [0020]It is directed to the hard disk driver 4 that the encryption software 5 corrects a file date by the file attribute of the movement destination saved at the hard disk 3 (STEP9). When a move original file is enciphered and it writes in a movement destination file, this is performed in order to prevent the date of a file from being updated. And the encryption software 5 takes out directions to the hard disk driver 4 so that a move original file may be deleted (STEP10), and a move original file is eventually deleted from the hard disk 3.

[0021]Thus, since a move original file is reproduced by the movement destination file after being enciphered automatically, and also a move original file is deleted by a series of processings of STEP 1-10 shown in drawing 2, it means that it was processed as a file was enciphered and it was moving. When moving a file to a non-code field from a code field, a move original file is similarly processed as it is decoded automatically and is moving. And the encryption software 5 returns the completion response of a move to OS (file system 1), after all the processings are completed.

[0022]Here, except [all] encryption / processing to decode of STEP5, the encryption software 5 gives directions to the hard disk driver 4, and is performing processing. However, in order to carry out direct access to the hard disk driver 4, without the encryption software 5 passing OS or the file system 1 and to enable it to perform STEP 1-10, the encryption software 5 must acquire the control function to the hard disk driver 4 beforehand. The procedure which acquires a control function required in order to perform each STEP of drawing 2 hereafter is

explained.

[0023]First, a procedure required in order that drawing 2 may perform STEP1 is explained. Drawing 3 shows the procedure for reading the attribute of a move original file. The fictitious file (here, it is considered as file name 'X') which gave the suitable name a priori is set to the encryption software 5. The encryption software 5 is directed to the hard disk driver 4 (hard disk 3) via OS and the file system 1 so that the attribute of a fictitious file may be read.

[0024]Here, since said fictitious file does not actually exist on the hard disk 3, the processing which reads an attribute serves as an error and processing is ended. However, since the control function for performing read-out and grant of an attribute from the file system 1 to the hard disk driver 4 will be sent out if processing which reads this attribute is performed even once, this is acquired and held with the encryption software 5. Henceforth, it can direct now from the encryption software 5 with a control function to the hard disk driver 4 directly, without passing the file system 1. Even if the thing of the same file name as said fictitious file exists on a hard disk also by chance, it does not change to a control function being sent out only by read-out of an attribute being successful. Therefore, the control function which performs read-out and grant of the attribute of a moved material in the procedure explained above is acquirable.

[0025]Next, the procedure which acquires the control function required in order to perform STEP2 of drawing 2 which opens a file is explained. Drawing 4 shows the procedure which acquires the control function which carries out file opening. The fictitious file (here, it is considered as file name 'X') which gave the suitable name a priori is set to the encryption software 5. First, the encryption software 5 accesses the hard disk driver 4 (hard disk 3) via OS and the file system 1, specifies the route of a drive, and looks for the element (a file or a folder) in it. And when a file does not find one, said fictitious file is directed to method OS of opening (it opens). When said fictitious file is read and it opens by dedicated mode at this time, since a fictitious file is no longer created and it does not consume a memory, it is preferred.

[0026]Here, since said fictitious file does not actually exist on the hard disk 3, the processing which opens a file serves as an error and processing is ended. However, if processing which opens this file is performed even once, the control function for opening a file from a file system to the hard disk driver 4 will be sent out. This is acquired and held with the encryption software 5. Henceforth, it can direct now from the encryption software 5 with a control function to the hard disk driver 4 directly, without passing the file system 1.

[0027]As a result of specifying the route of a drive and looking for the element (a file and a folder) in it, when a suitable folder or file is found, It directs to consider that the file which it pointed to OS so that it might consider that the found folder is a file and it might be opened, or was found is a folder, and to open the fictitious file X (file which does not exist actually) in the folder to OS. In any case, if it does in this way, a processing result will serve as an error, but

the control function for opening a file from the file system 1 to the hard disk driver 4 similarly is sent out.

[0028]If the control function which opens a file in the above-mentioned procedure is acquired, the control function which reads a file in the process, the control function written in a file, and the control function which closes a file are simultaneously acquirable. Therefore, a control function required in order to perform STEP 4, 6, 7, and 8 of drawing 2 is also acquirable by acquiring the control function which opens a file.

[0029]Next, the procedure which acquires the control function required in order to perform STEP10 of drawing 2 which deletes a file is explained. Drawing 5 shows the procedure which acquires the control function which deletes a file. The fictitious file (here, it is considered as file name 'X') which gave the suitable name a priori is set to the encryption software 5. First, the encryption software 5 accesses the hard disk driver 4 (hard disk 3) via OS and the file system 1, specifies the route of a drive, and looks for the element (a file or a folder) in it. And when a file does not find one, it is directed to OS that said fictitious file deletes from the hard disk 3. At this time, since said fictitious file does not exist in the hard disk 3, it becomes an error and processing is ended. However, if processing which deletes a file is performed even once, the control function for deleting a file from the file system 1 to the hard disk driver 4 will be sent out. This is acquired and held with the encryption software 5. Henceforth, it can direct now from the encryption software 5 with a control function to the hard disk driver 4 directly, without passing a file system.

[0030]As a result of specifying the route of said drive and looking for the element (a file or a folder) in it, when a folder or a file is found, it directs to consider that the found folder is a file and to delete it to OS. Or it directs to consider that the found file is a folder and to delete fictitious file 'X' in the folder to OS. Although a processing result serves as an error and ends processing with any directions, the control function for deleting a file from the file system 1 to the hard disk driver 4 similarly will be sent. It becomes possible to acquire a control function above required in order to perform STEP1-STEP4 of drawing 2, and STEP 6-10.

[0031]When a predetermined folder is chosen as an object of user's move origin, the file which exists in it, and the file in a subfolder and also the subfolder concerned will also be moved as well as said folder as a moved object. In this case, as that example is shown in drawing 6, it is common to perform moving processing in the procedure what is called based on the recursive technique. Since it always memorizes which file or folder this disposal method moved during moving processing, and there is nothing if it is kana ***** , a considerable quantity of main memory will be consumed.

[0032]Then, the moving processing method (henceforth the nonreflexive technique) of the folder which stopped the amount of consumption of main memory in this invention is proposed. The example of composition of a concrete source folder is given to drawing 7, and this is

explained to it in detail. In drawing 7, O seal shows a folder and ** seal shows the file. Now, the top folder of a moved material is assumed to be the folder 1, and the file 1, the file 2, and the subfolder (folders 2 and 5) exist in it. Two or more subfolders and files exist under the folder 2, and five folders (the folder 1 - the folder 5) and eight files (files 1-8) exist as a whole. [0033]First, it is directed to the hard disk driver 4 that encryption software reads the attribute of the top folder (folder 1) of a source folder. And based on the information on said read attribute, the duplicate of the folder 1 is created in a movement destination, and it is considered as a movement destination folder (movement destination folder 1). If an attribute is furthermore given to the movement destination folder 1 (correction of a day entry), it will return to the folder 1 and the element (a file and the general term of a subfolder are said) of the beginning in the folder 1 will be looked for. It is made to move into the movement destination folder 1, although the files 1 and 2 and the folders 2 and 5 exist in the folder 1, enciphering or decoding the file 1 and the file 2 first in the procedure shown in drawing 2, respectively. It will be deleted from the folder 1 which is a source folder by the file 1 and the file 2 which finished moving processing at this time.

[0034]In this state, one file does not exist directly under the folder 1, either. Then, it is if a file will be in the state where one does not exist. It moves to the folder 2 under it, and the attribute is read by making this into a new source folder. Based on the attribute of said read folder 2, the duplicate of the folder 2 is similarly created in a movement destination, and it is considered as a movement destination folder (movement destination folder 2). If an attribute is furthermore given to the movement destination folder 2 (correction of a day entry), it will return to the folder 2 and the element in the folder 2 will be looked for. Although the files 3 and 4 and the folders 3 and 4 are found as an element in the folder 2, when a file is found, the file 3 and the file 4 are moved into the movement destination folder 2, enciphering or decoding in the procedure similarly shown in drawing 2. It will be deleted from the folder 2 which is a source folder by the file 3 and the file 4 which finished moving processing at this time. If a subfolder is found as an element in the folder 2 (here, the folders 3 and 4 correspond), next, it will move to a subfolder and the same procedure as the following will be repeated.

[0035]Here, it moves to the folder 3 first and the attribute is read by making this into a new move original file. Based on the attribute of said read folder 3, the duplicate of the folder 3 is created in a movement destination, and it is considered as a movement destination folder (movement destination folder 3). And if an attribute is given to the movement destination folder 3 (correction of a day entry), it will return to the folder 3 and the element in the folder 3 will be looked for. Only the file 5 and the file 6 exist in the folder 3. When a file exists, the file 5 and the file 6 are moved to the movement destination folder 3, enciphering or decoding in the procedure similarly shown in drawing 2. The file 5 and the file 6 which finished moving processing at this time are in the state where it was deleted from the folder 3 which is a source

folder. Since the inside of the after-movement folder 3 will be in the state where nothing exists as an element from the inside of the folder 3 about the files 5 and 6, the folder 3 which became empty from the moved material is deleted.

[0036]Next, when the eliminated folder does not correspond to the top folder (folder 1), it once returns to the top folder (folder 1), and the above-mentioned procedure is again repeated from the top folder 1. The procedure is explained below. Now, it is in the state where the folder 1, the folder 2, the folder 4, the folder 5, and the files 7 and 8 exist in the moved material. Other files and folders are already deleted from the moved material.

[0037]First, if the element in the folder 1 is looked for, the folder 2 will be found as a subfolder. The folder 2 is carried out new move origin, and the attribute of the folder 2 is read. Although the duplicate of the folder 2 is created in a movement destination below, since it is already ending with creation. The element of the beginning in the folder 2 of a movement destination is looked for. Only the folder 4 exists in the folder 2. Then, the processing to look for is ended and the folder 4 is carried out new move origin.

[0038]Next, the attribute of the folder 4 is read, and also the duplicate of the folder 4 is created in a movement destination based on the attribute of said read folder 4, and it is considered as the movement destination folder 4. And if an attribute is given to the movement destination folder 4 (correction of a day entry), it will return to the folder 4 and the element in the folder 4 will be looked for. Only the files 7 and 8 exist in the folder 4. the procedure similarly shown in drawing 2 when a file exists -- a code fault -- or the file 7 and the file 8 are moved to the movement destination folder 4, decoding. The file 7 and the file 8 which finished moving processing at this time are in the state where it was deleted from the folder 4 which is a source folder. Since the inside of the after-movement folder 4 will be in the state where nothing exists as an element from the inside of the folder 4 about the files 7 and 8, the folder 4 which became empty from the moved material is deleted.

[0039]Next, when the eliminated folder does not correspond to the top folder (folder 1), it returns to the top folder (folder 1), and the again same procedure is repeated from the top folder 1. The procedure is explained below.

[0040]Now, it is in the state where the folder 2 and the folder 5 exist in the moved material. Other files and folders are already deleted from the moved material. If the procedure is followed below, the folder 2 will turn into a source folder, but since the inside of the folder 2 serves as empty, it is deleted, and it returns with the top folder (folder 1). Next, since the folder 5 exists as a subfolder in the folder 1, this serves as a source folder. The duplicate of the folder 5 is created in a movement destination, and since the inside of the folder 5 is empty, it deletes the folder 5 from a moved material. Therefore, all the files and subfolder in the folder 1 of a moved material were moved to the movement destination in the procedure explained above. The folder 1 of the move origin which is finally the top folder is deleted, and movement is

completed. As mentioned above, if the procedure explained using drawing 7 is generalized, it will become as the flow chart shown in drawing 8.

[0041]Next, the procedure required in order to perform each procedure for the bottom shown in drawing 7 and 8 moving a folder which acquires a control function is explained. Drawing 9 shows the procedure which acquires the control function which looks for the element of the beginning of a source folder. It is directed to OS that the encryption software 5 specifies the route of a drive and looks for the element (a folder and a file) of the beginning in it. Even if it is not found even if an element is found as a result or, as a result of making OS perform the above-mentioned processing, the control function for looking for the first element from the file system 1 to the hard disk driver 4 is sent out. This is acquired and held with the encryption software 5. Henceforth, it can direct now from the encryption software 5 with a control function to the hard disk driver 4 directly, without passing the file system 1. As a result of acquiring the control function which looks for said first element, the control function which looks for the following element can be obtained.

[0042]Next, the procedure which acquires the control function which creates or deletes a source folder is explained. Drawing 10 shows the procedure which acquires the control function which creates or deletes a source folder. The fictitious file (here, it is considered as file name 'X') which gave the suitable name a priori is set to the encryption software 5. First, the encryption software 5 accesses the hard disk driver 4 (hard disk 3) via OS and the file system 1, specifies the route of a drive, and looks for the element (a file or a folder) in it.

[0043]And when neither a file nor a folder is found, it directs to delete said fictitious file 'X' from the hard disk 3 to OS. When a file or a folder is found, it directs to create the folder of the same name as the found file or folder to OS. Since the file or folder of a same name already exists as a result, it becomes improper to create a folder under the same name, and processing serves as an error and is ended. However, the control function which creates or deletes a folder is acquirable by making OS perform the above-mentioned processing.

[0044]Since it becomes possible for the encryption software 5 to acquire a required control function via the file system 1 at the time of starting, and to control the hard disk driver 4 directly by the procedure explained above using this, It becomes possible to carry out processing which moves a move original file or a folder to a movement destination while performing encryption/decoding. And when all the processings were completed, it enabled it to return a shift request response to the file system 1 by using a control function for the hard disk driver 4, and having made it direct directly from the encryption software 5. Therefore, it becoming impossible to access a file, making a mistake in the file under processing, and also making it move to somewhere else can be prevented during processing. Therefore, the 3rd person can be prevented also from that a user also mistakes with what finished processing thru/or acquiring ***** unjustly, even if a user does a leaving chair during processing in spite of being

under processing. The 3rd person invades into a user's personal computer via a network, and acquiring the file under encryption processing unjustly can also be prevented. As mentioned above, this invention is applicable also to a duplicate, although movement of a file etc. was explained to the example.

[0045]Next, when encryption software is started, how to generate key data required for encryption or decoding is explained. Drawing 11 is a block diagram showing the process in which a common encryption key or decode key is generated. If encryption software is started as shown in drawing 11, the display which stimulates the input of ID (identification signal) on a monitor first will be made, and a user will input ID from a keyboard according to this. The numerals of the predetermined bit which performs a hash function operation based on this, and is called a hash value are generated, and also a predetermined encryption key is generated based on a hash value. The generated encryption key is saved at a hard disk, and when it enciphers or decodes a predetermined electronic file, it is used.

[0046]However, in this encryption key generation method, since the encryption key is saved at the hard disk, the 3rd person invades into a user's computer via a network, and it has the fault of being easy to acquire an encryption key from a hard disk unjustly. Then, in order to cancel this fault, the generation method of a new encryption key is shown in drawing 12. In drawing 12, a user inputs ID from a keyboard according to the display to which the ID input on a monitor is urged. A hash function operation is performed based on this, and the hash value (message digest) of a predetermined bit is generated. Next, said hash value is divided into two, one side is set to hash value A, and another side is set to hash value B. Said hash value is divided into multiple-data-stream, two with said arbitrary multiple-data-stream are chosen here, and it is good also as said hash value A and hash value B.

[0047]So, when generate an encryption key based on said hash value A, making this hold only to main memory and terminating software, an encryption key is deleted from main memory. On the other hand, said hash value B is saved as authentication data at a hard disk. And if the authentication data generated when a user inputs ID from a computer next time is compared with said saved authentication data and authentication data is in agreement, an encryption key will be generated based on said hash value A, and it will enable it to carry out desired encryption or decoding. When authentication data is not in agreement, the generated encryption key is canceled and it prevents from performing encryption or decoding processing.

[0048]Since the encryption key generated from hash value A is not saved on a hard disk if it does in this way, it becomes impossible for the 3rd person to obtain this unjustly via a network. and a user -- since only the person himself/herself cannot start encryption software, it becomes an effective means, when it also becomes impossible of the 3rd person to decode an enciphered file unjustly on a user's computer and he secures security.

[0049]

[Effect of the Invention]As this invention was explained above, when encryption software performs file migration processing accompanied by encryption or decoding conventionally, In order to have to control a hard disk driver (hard disk) via OS, A problem when carrying out moving processing accompanied by said encryption or decoding after returning the completion response of a move which shows that movement of a file was completed to OS, The problem which is easy to acquire the generated encryption key from the 3rd person unjustly is solved, Have a procedure which acquires the control function with which encryption software controls a hard disk driver (hard disk) directly, and perform file migration processing accompanied by encryption or decoding using said control function which carried out income, and. It is **** about higher efficacy for processing stability supplying movement of the data of encryption software with a high security function, a duplicating method and encryption, and a decoding method highly, since the generated encryption key is held only on main memory and it was made not to save it on a hard disk.

[0050]

[Translation done.]

* NOTICES *

JP0 and INPIT are not responsible for any damages caused by the use of this translation.

- 1.This document has been translated by computer. So the translation may not reflect the original precisely.
- 2.**** shows the word which can not be translated.
- 3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1]The block diagram showing the composition of the encryption software concerning this invention.

[Drawing 2]The flow chart figure showing file migration processing of the encryption software concerning this invention.

[Drawing 3]The flow chart figure showing the procedure which acquires the control function in which the encryption software concerning this invention carries out attribute read-out and attribute grant.

[Drawing 4]The flow chart figure showing the procedure which acquires the control function which opens the file of the encryption software concerning this invention.

[Drawing 5]The flow chart figure showing the procedure which acquires the control function which deletes the file of the encryption software concerning this invention.

[Drawing 6]The flow chart figure by the recursive technique which showed the folder moving processing of the encryption software concerning this invention.

[Drawing 7]The figure showing an example of the source folder of the encryption software concerning this invention.

[Drawing 8]The flow chart figure by the nonreflexive technique which showed the folder moving processing of the encryption software concerning this invention.

[Drawing 9]The flow chart figure showing the procedure which acquires the control function which looks for the element of the beginning of the encryption software concerning this invention.

[Drawing 10]The flow chart figure showing the procedure which acquires the control function which creates the folder of the encryption software concerning this invention, and is deleted.

[Drawing 11]The flow chart figure showing the 1st method of generating the encryption key of the encryption software concerning this invention.

[Drawing 12]The flow chart figure showing the 2nd method of generating the encryption key of the encryption software concerning this invention.

[Drawing 13]The block diagram showing the composition of the conventional encryption software.

[Drawing 14]The flow chart figure showing file migration processing of the conventional encryption software.

[Description of Notations]

- 1 ... File system
- 2 ... Operation system
- 3 ... Hard disk
- 4 ... Hard disk driver
- 5 ... Encryption software

[Translation done.]

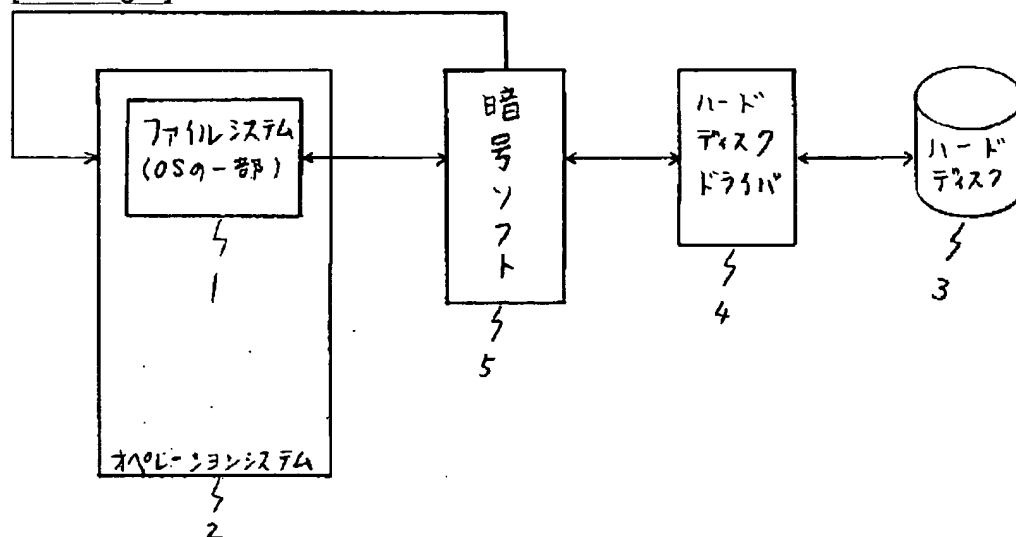
* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

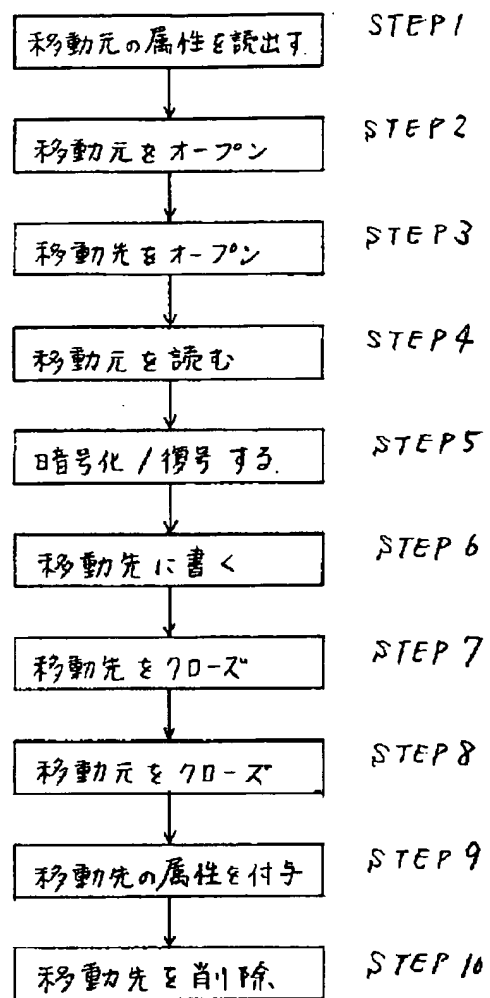
- 1.This document has been translated by computer. So the translation may not reflect the original precisely.
- 2.**** shows the word which can not be translated.
- 3.In the drawings, any words are not translated.

DRAWINGS

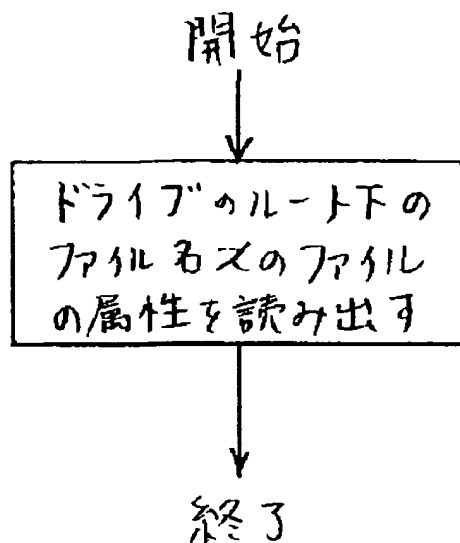
[Drawing 1]



[Drawing 2]



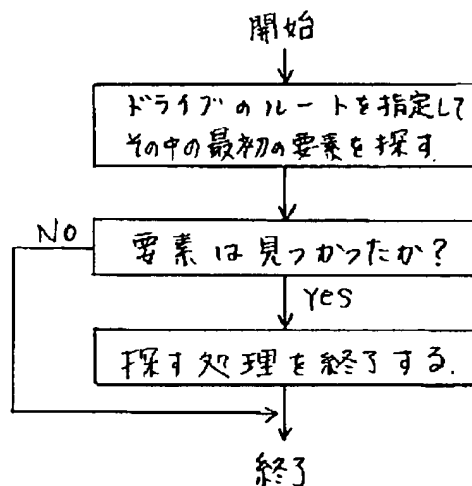
[Drawing 3]



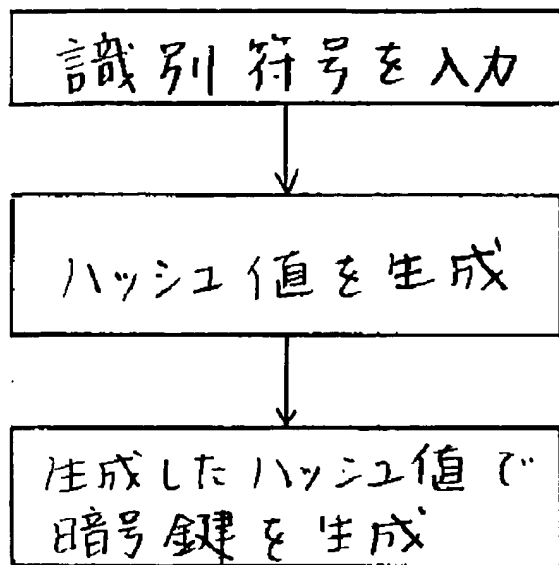
属性読み出し及び属性を付与
する制御関数を取得する手順

[Drawing 9]

最初の要素を探す制御関数を取得する手順

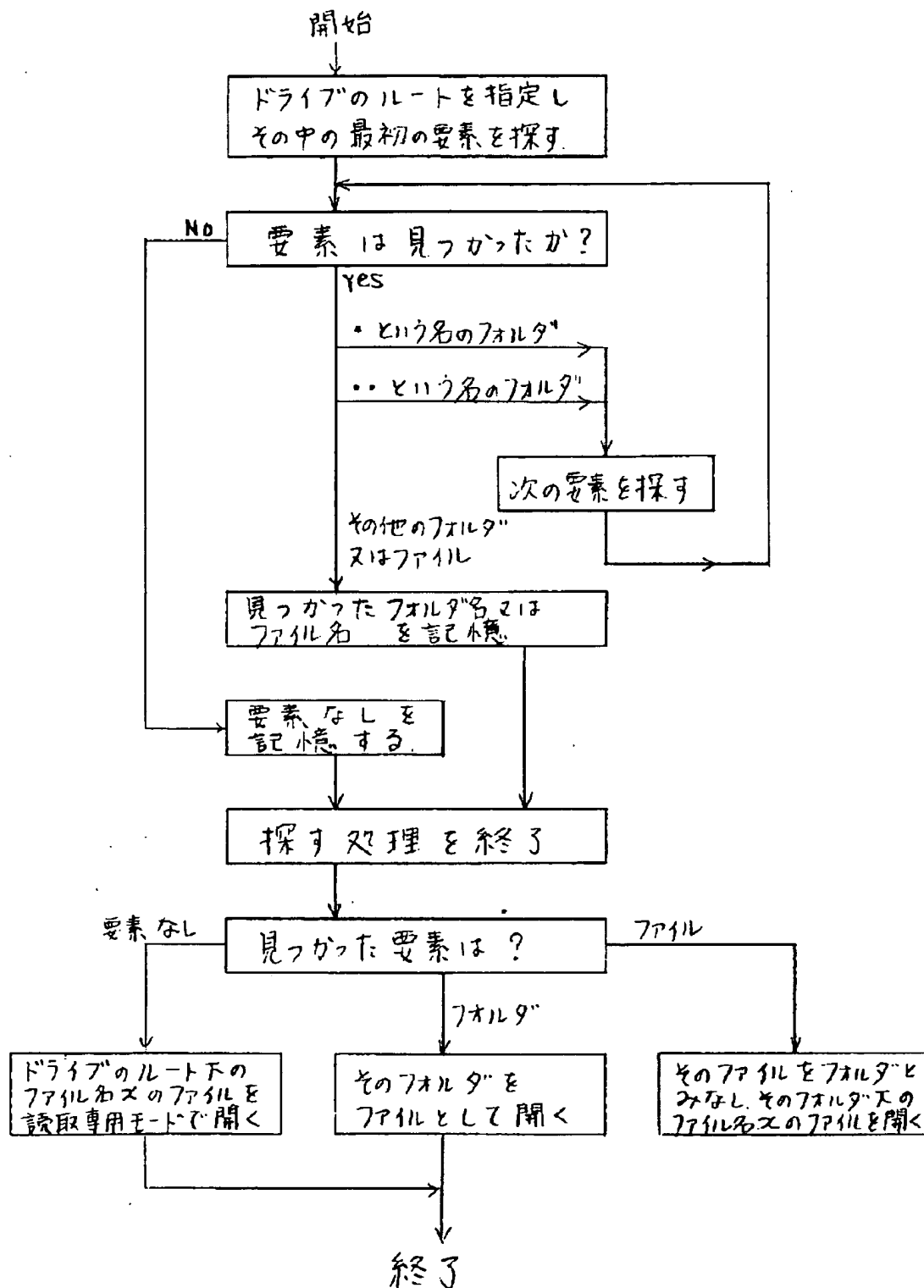


[Drawing 11]



[Drawing 4]

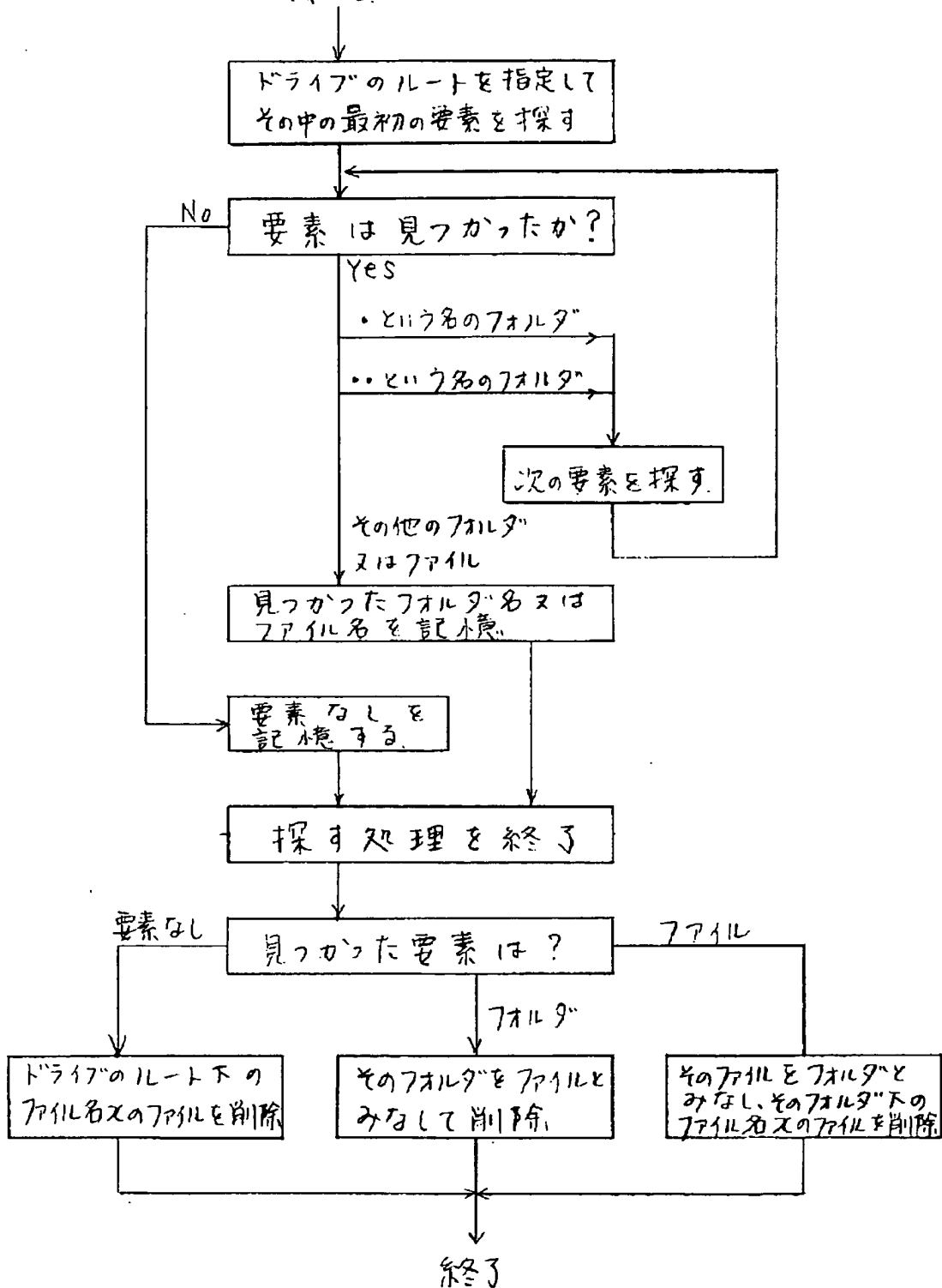
ファイルをオープンする制御関数を取得する手順



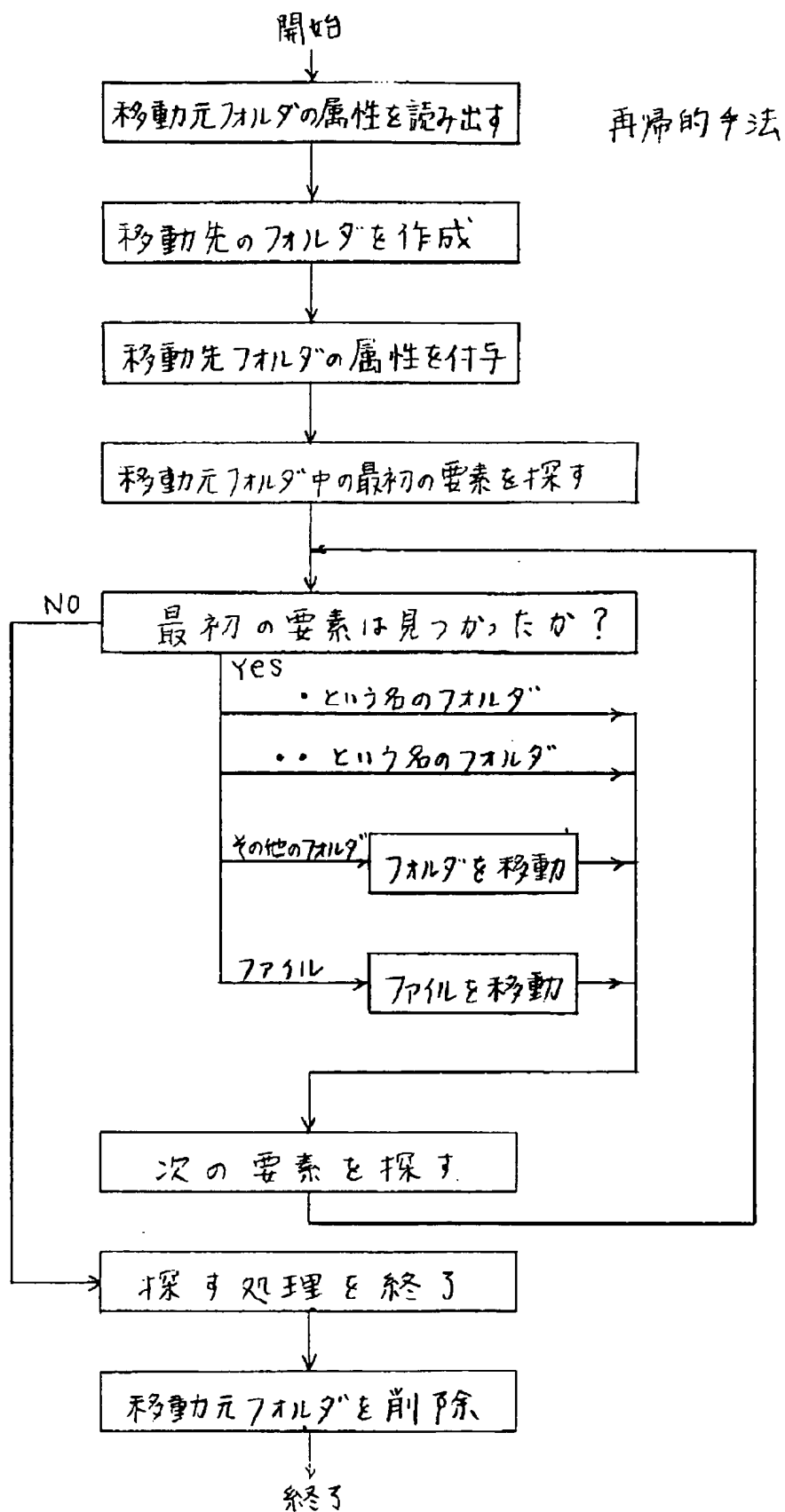
[Drawing 5]

ファイルを削除する制御関数を取得する手順

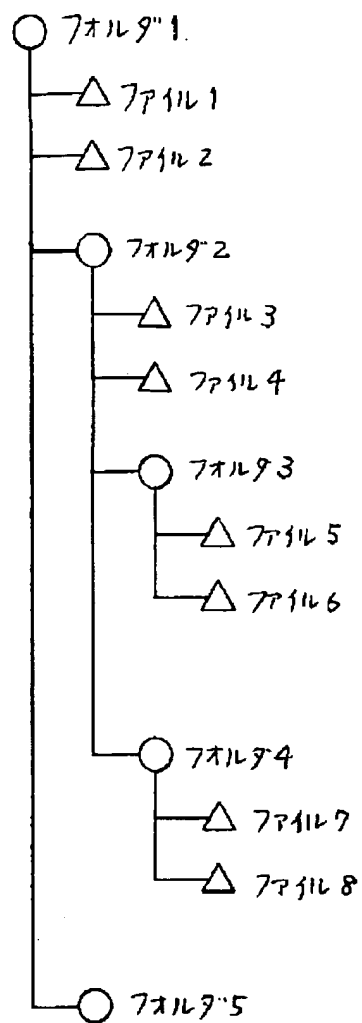
開始



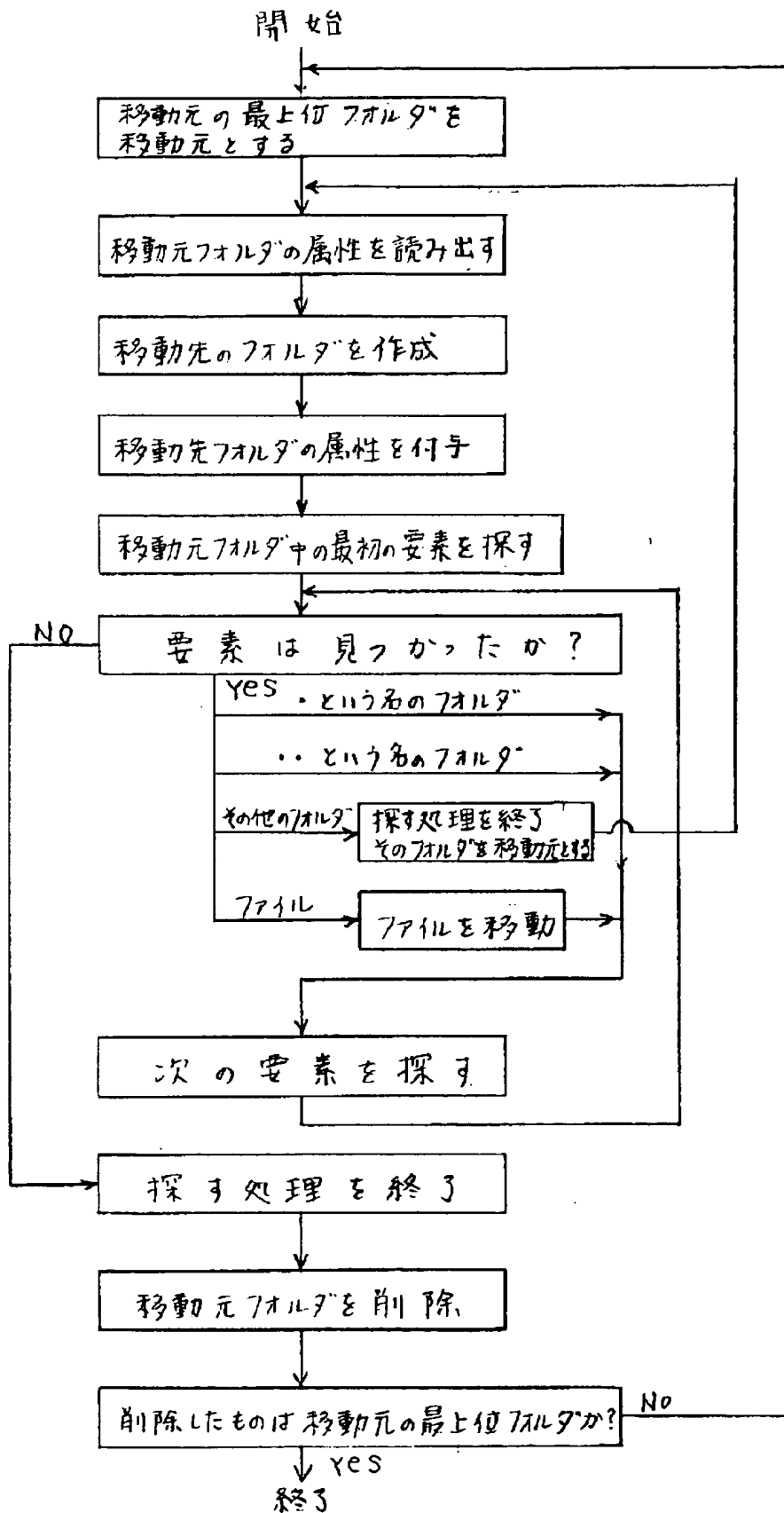
[Drawing 6]



[Drawing 7]

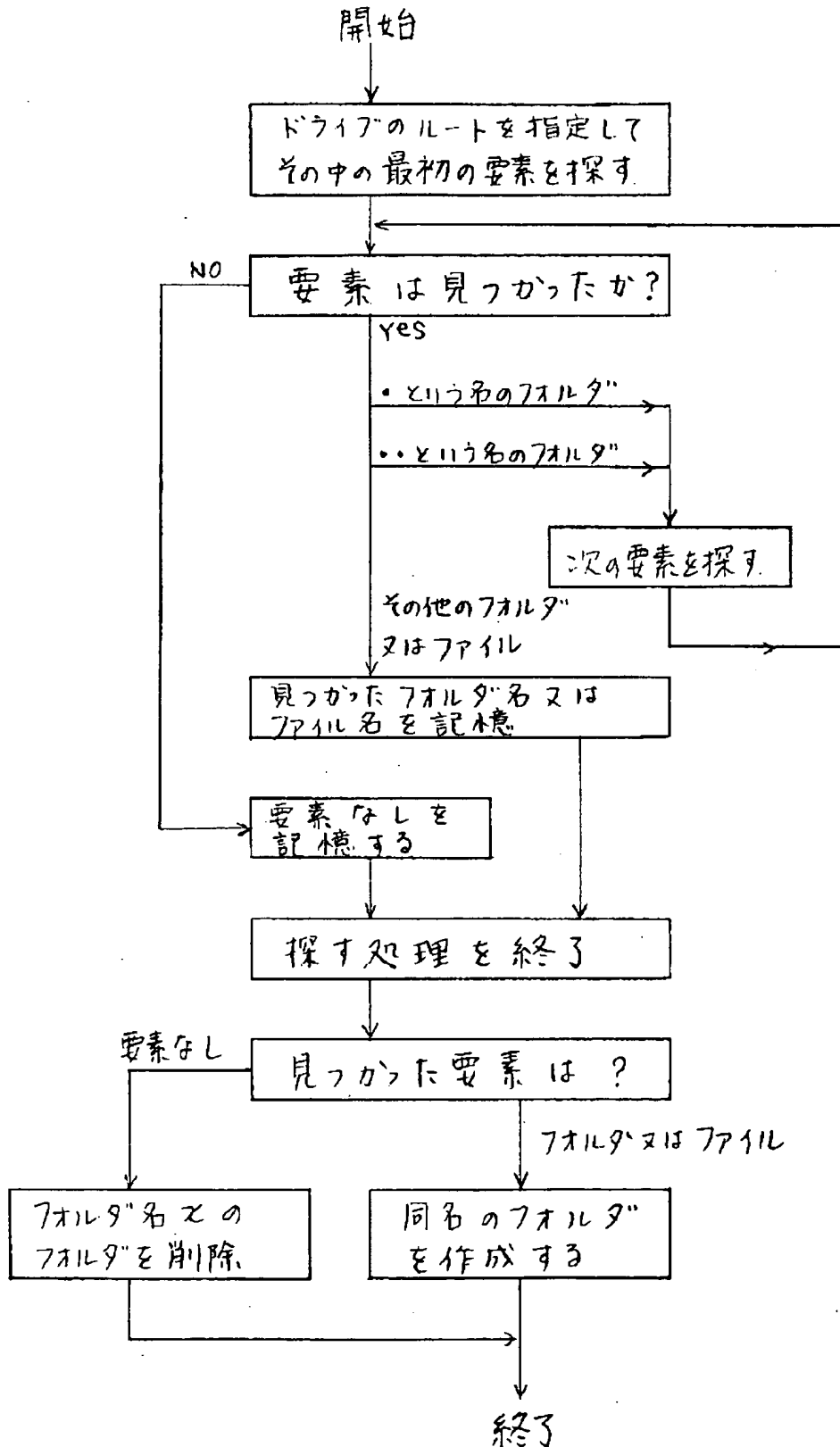


[Drawing 8]

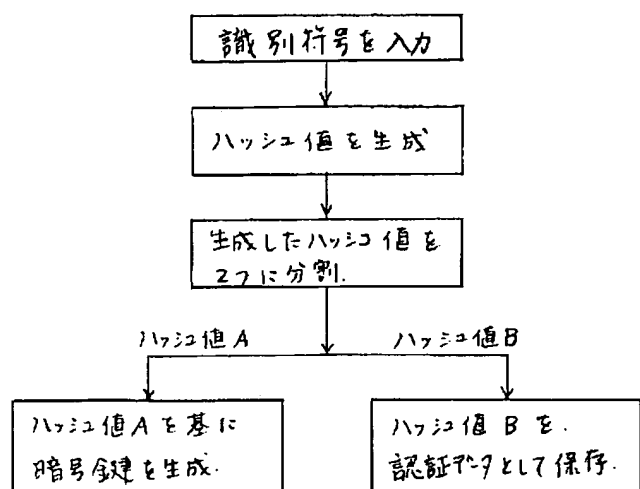


[Drawing 10]

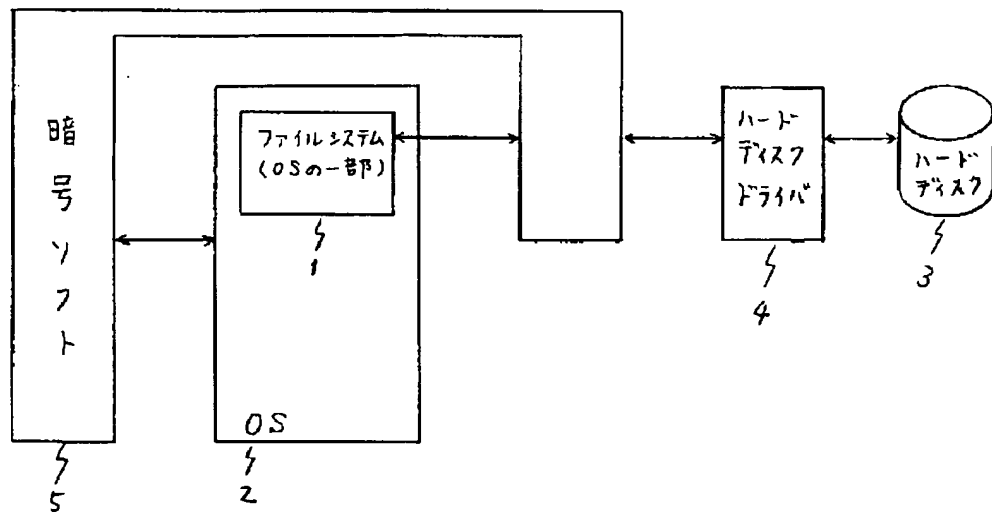
フォルダを作成,削除する制御関数を取得する手順



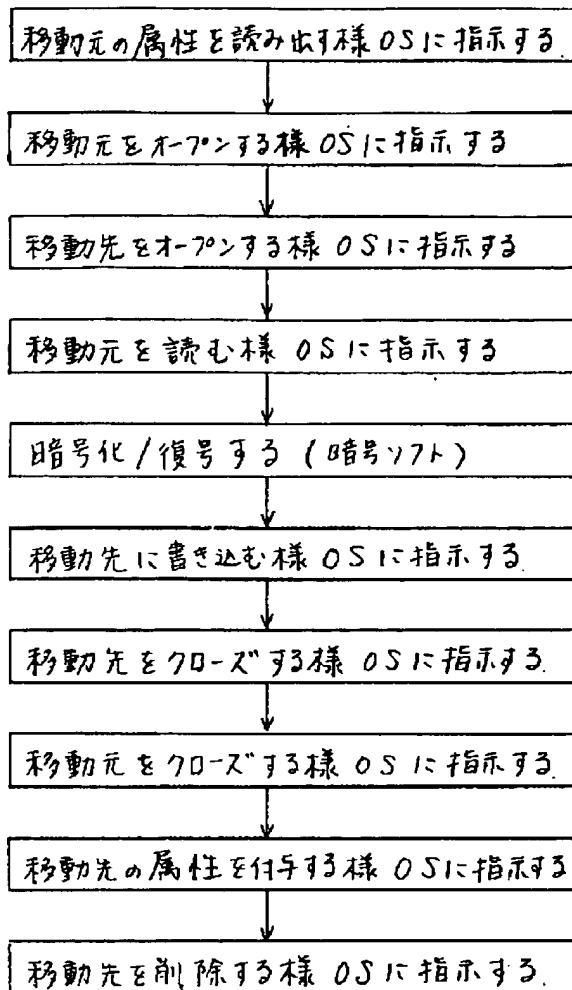
[Drawing 12]



[Drawing 13]



[Drawing 14]



[Translation done.]